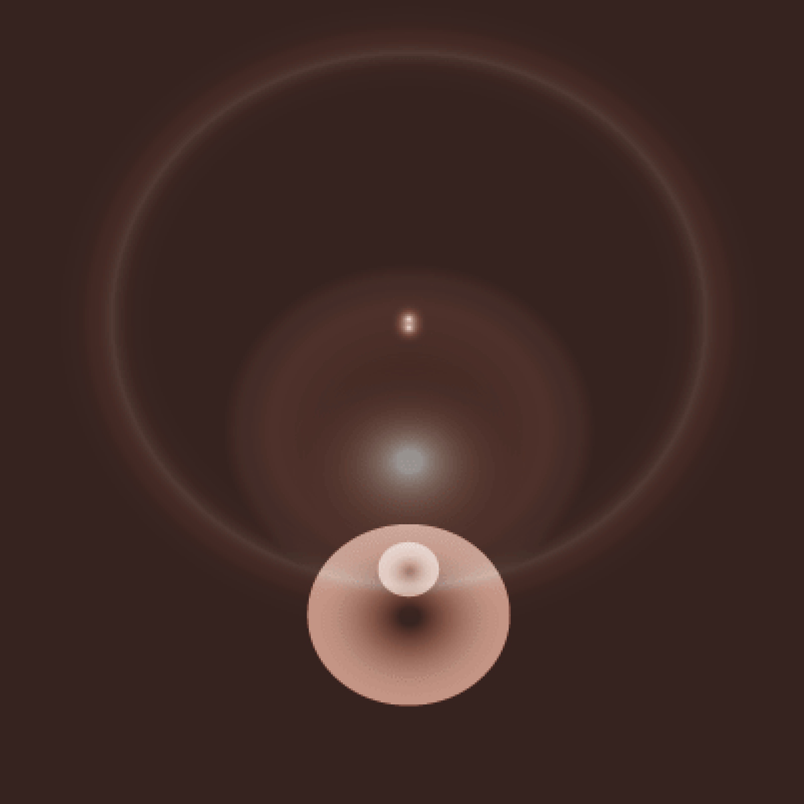


沈蕴梅 童文俊 符钰 主编
郑广成 顾蓬蓬 沈晔 副主编

ASP.NET动态Web开发技术项目化教程

21世纪计算机科学与技术实践型教程

丛书主编 陈明

清华大学出版社

21世纪计算机科学与技术实践型教程

ASP.NET动态Web开发技术项目化教程

沈蕴梅　童文俊　符　钰　主　编

郑广成　顾蓬蓬　沈　晔　副主编

清华大学出版社

北　京

内容简介

ASP.NET是基于.NET平台开发动态Web应用程序的技术，它是.NET框架的一部分。本书采用C#语言讲解ASP.NET的核心技术，由浅入深、循序渐进，采用典型的项目载体，采取课内外项目并行、工作过程项目化的模式，系统地介绍ASP.NET系统架构与逻辑设计、实体设计与多层依赖、数据库访问与实现、动态Web界面设计与编码、Excel高级报表、数据查询统计等高级编码技术和方法。为了便于读者全面掌握程序设计技术和规范，深刻体会编程的乐趣，最后给出一个综合性的实战项目，全面讲述了Web应用系统的开发全过程。

本书是作者在多年实践教学过程中总结提炼而成的，采用了“工作过程项目化”的教学流程进行内容重组，通过示例讲解知识点和技能点，课内主训一个项目，课外并行实战一个项目，通过项目导入（导入课内训练项目的项目场景、引导问题）、技术与知识准备（讲、练“课内主讲项目或示例”）、项目训练（完成课内训练项目）、平行项目训练（完成平行项目）的流程实现本书单元内容编写。

为方便教学，本书可提供教学课件和各单元源代码程序。

本书可作为高职高专、应用型本科院校相关专业的教材，也可作为编程爱好者的参考书。

图书在版编目（CIP）数据

ASP.NET动态Web开发技术项目化教程/沈蕴梅等主编. —北京：清华大学出版社，2016
21世纪计算机科学与技术实践型教程
ISBN 978-7-302-43135-0

Ⅰ.①A…　Ⅱ.①沈…　Ⅲ.①网页制作工具－程序设计－高等学校－教材　Ⅳ.①TP393.092

中国版本图书馆CIP数据核字（2016）第050156号

责任编辑：谢　琛
封面设计：何凤霞
责任校对：白　蕾
责任印制：刘海龙

出版发行：清华大学出版社
网　　址：http://www.tup.com.cn，http://www.wqbook.com
地　　址：北京清华大学学研大厦A座　　**邮　　编**：100084
社 总 机：010-62770175　　**邮　　购**：010-62786544
投稿与读者服务：010-62776969，c-service@tup.tsinghua.edu.cn
质量反馈：010-62772015，zhiliang@tup.tsinghua.edu.cn
课件下载：http://www.tup.com.cn，010-62795954
印 刷 者：三河市君旺印务有限公司
装 订 者：三河市新茂装订有限公司
经　　销：全国新华书店
开　　本：185mm×260mm　　**印　　张**：13　　**字　　数**：298千字
版　　次：2016年4月第1版　　**印　　次**：2016年4月第1次印刷
印　　数：1～2000
定　　价：29.00元

产品编号：066057-01

《21世纪计算机科学与技术实践型教程》

编辑委员会

《21 世纪计算机科学与技术实践型教程》

序

21 世纪影响世界的三大关键技术：以计算机和网络为代表的信息技术；以基因工程为代表的生命科学和生物技术；以纳米技术为代表的新型材料技术。信息技术居三大关键技术之首。国民经济的发展采取信息化带动现代化的方针，要求在所有领域中迅速推广信息技术，导致需要大量的计算机科学与技术领域的优秀人才。

计算机科学与技术的广泛应用是计算机学科发展的原动力，计算机科学是一门应用科学。因此，计算机学科的优秀人才不仅应具有坚实的科学理论基础，而且更重要的是能将理论与实践相结合，并具有解决实际问题的能力。培养计算机科学与技术的优秀人才是社会的需要、国民经济发展的需要。

制订科学的教学计划对于培养计算机科学与技术人才十分重要，而教材的选择是实施教学计划的一个重要组成部分，《21 世纪计算机科学与技术实践型教程》主要考虑了下述两方面。

一方面，高等学校的计算机科学与技术专业的学生，在学习了基本的必修课和部分选修课程之后，立刻进行计算机应用系统的软件和硬件开发与应用尚存在一些困难，而《21 世纪计算机科学与技术实践型教程》就是为了填补这部分空白。将理论与实际联系起来，使学生不仅学会了计算机科学理论，而且也学会了应用这些理论解决实际问题。

另一方面，计算机科学与技术专业的课程内容需要经过实践练习，才能深刻理解和掌握。因此，本套教材增强了实践性、应用性和可理解性，并在体例上做了改进——使用案例说明。

实践型教学占有重要的位置，不仅体现了理论和实践紧密结合的学科特征，而且对于提高学生的综合素质，培养学生的创新精神与实践能力有特殊的作用。因此，研究和撰写实践型教材是必需的，也是十分重要的任务。优秀的教材是保证高水平教学的重要因素，选择水平高、内容新、实践性强的教材可以促进课堂教学质量的快速提升。在教学中，应用实践型教材可以增强学生的认知能力、创新能力、实践能力以及团队协作和交流表达能力。

实践型教材应由教学经验丰富、实际应用经验丰富的教师撰写。此系列教材的作者不但从事多年的计算机教学，而且参加并完成了多项计算机类的科研项目，他们把积累的经验、知识、智慧、素质融于教材中，奉献给计算机科学与技术的教学。

我们在组织本系列教材过程中，虽然经过了详细的思考和讨论，但毕竟是初步的尝试，不完善甚至缺陷不可避免，敬请读者指正。

本系列教材主编　陈明

2005 年 1 月于北京

前　言

微软公司推出的 ASP.NET 技术，以其成功的发展战略和技术本身的无限魅力，在短短几年的时间里取得骄人的成绩，受到行业专业人士和学习者的青睐。

本书主要基于岗位技能、软件流程和规范，采取了"工程过程项目化"的编写模式进行编写的，是教学团队结合"产教融合、产学并行"的教学改革和实践总结出来的教学模式和教学内容的展现，团队经过筛选和提炼后，确定典型项目作为教学内容载体，更适合以应用能力为本位的高职高专和应用本科的教学及训练的要求。

该书通过项目导入提出问题，通过技术与知识准备，解决问题并掌握相应的技术和方法，然后回到项目训练完成项目，再通过并行项目进一步训练，达到巩固和举一反三的训练效果，实现了课内外项目并行推进的教学形式。本书打破了传统的学科章节和硬项目化编写形式，采取了产学并行的形式进行内容组编，全书共分十章，最后一章通过综合项目训练学生技能，进一步提高学生的应用实践能力，体现了"做中学、学中产"的实训教学思想。该书主要内容如下：

第 1 章 ASP.NET 系统架构与逻辑设计。

第 2 章 实体设计与多层依赖。

第 3 章 数据库访问与实现。

第 4 章 动态 Web 界面设计与编码。

第 5 章 数据绑定与高级编码。

第 6 章 WebService 和 Ajax 应用。

第 7 章 数据导入、导出与报表。

第 8 章 系统优化与测试。

第 9 章 系统发布与部署。

第 10 章 综合项目实训。

本书由沈蕴梅、童文俊、符钰主编，郑广成、顾蓬蓬、沈晔副主编，沈蕴梅负责统稿，何光明、王珊珊、卢振侠、石雅琴、曹冬梅、陈莉萍等参与了部分章节的校对工作。在本书的编写过程，得到了苏州吉耐特信息科技有限公司孔小兵、江苏微软技术中心朱新立两位工程师的参与和资源支持，属于一本校企合作教材。该书根据技术模块设置单元，根据典型项目设计内容载体，通过课内外两个项目并行推进来提高学生的应用能力和创新能力，具有实战性、可操作性、新颖性、通俗性和项目过程化的特点，更加激发学生学习兴趣和主动

性。为方便教学，本书配有教学课件和各单元源代码程序，读者可登录清华大学出版社网站下载。

由于时间仓促，再加上编者水平有限，书中难免有错误和疏漏之处，敬请广大读者批评指正。

作　者

2016年2月

目　　录

第1章　ASP.NET系统架构与逻辑设计

本章要点：

- 搭建 ASP.NET 开发环境
- 搭建系统架构
- 创建数据库

技能目标：

- 会搭建系统架构
- 能绘制功能模块图，确定系统数据库

1.1　项目导入

【项目场景】

时光飞逝，我们即将离开美丽的校园。作为软件1114班的一员，小张对班级和同学都有着深厚的感情，他如何表达这种对班级的独特情感，如何在离开学校后还能留守这份真情呢？他决定用一个班级特有的网站，记录三年来班级里的点点滴滴，体现他对班级的情有独钟。

【问题引导】

(1) 如何搭建系统架构。
(2) 如何确定系统功能模块。
(3) 班级网站有哪些数据表。
(4) 用什么工具开发系统。

1.2　技术与知识准备

1.2.1　ASP.NET 介绍

目前开发动态网站的主要技术有 ASP.NET、JSP、PHP、ASP 等，其中 ASP.NET 是

基于.NET平台创建动态网页的一种服务器端技术，是基于B/S的应用程序，使用它可创建动态可交互的Web页面。在微软的.NET战略中，ASP.NET是其中的一项核心技术，是.NET Framework的重要组成部分。.NET Framework包括两个重要组件：.NET Framework类库和公共语言运行时，编写ASP.NET页面需要用到.NET Framework的框架类库和公共语言运行时。

ASP.NET、.NET Framework以及Visual Studio的版本一直以来都在不断地更新，在这些版本中，具有革命性意义的是ASP.NET 2.0、.NET Framework2.0以及Visual Studio2005，.NET Framework2.0的出现标志着.NET Framework真正走向成熟。ASP.NET 3.5在沿用了ASP.NET 2.0的优势下又加入了C＃3.0、LINQ、ASP.NET AJAX 3.5和REST等元素。ASP.NET 4.0、.NET Framework4.0以及Visual Studio2010是微软2010年4月发布的，在新的版本中，微软解决了只能提示不友好、Web部署复杂等诸多问题，并且添加了支持多显示器、支持TDD、内嵌本地JQuery等新特性。在ASP.NET 4.0中新增了SEO优化支持、ASP.NET MVC、QueryExtender服务器控件等。本书采用的是微软最新发布的ASP.NET 4.5、.NET Framework4.5以及Visual Studio2012，.NET Framework 4.5包括ASP.NET 4.5增强功能。Visual Studio 2012还包括增强功能和新功能的增强的Web开发。具体版本演进历程如表1.1所示。

表1.1 ASP.NET、.NET Framework、Visual Studio版本演进历史

序 号	Visual Studio版本	.NET Framework版本	ASP.NET版本
1	VS2002	1	1
2	VS2003	1.1	1.1
3	VS2005	2	2
4	VS2008	3/3.5	3.5
5	VS2010	4	4
6	VS2012	4.5	4.5

1.2.2 B/S系统搭建技术

【步骤1】创建解决方案。

使用解决方案可以将多个文件夹以树型结构方式组织成一体，使得项目结构清晰。选择“文件”→“新建”→“项目”，在弹出的新建项目中，展开已安装的模板中的“其他项目类型”节点，选择“Visual Studio解决方案”，并更改名称为“News”，如图1.1所示。

【步骤2】添加实体层。

(1) 右击解决方案，选择“添加”→“新建项目”，如图1.2所示。

(2) 在添加新项目的对话框中，在左边已安装的模板中选择Visual C＃，中间选择“类库”，然后为实体层命名，命名的一般规则是，实体层的项目命名为解决方案名＋Model，即“NewsModel”，如图1.3所示。

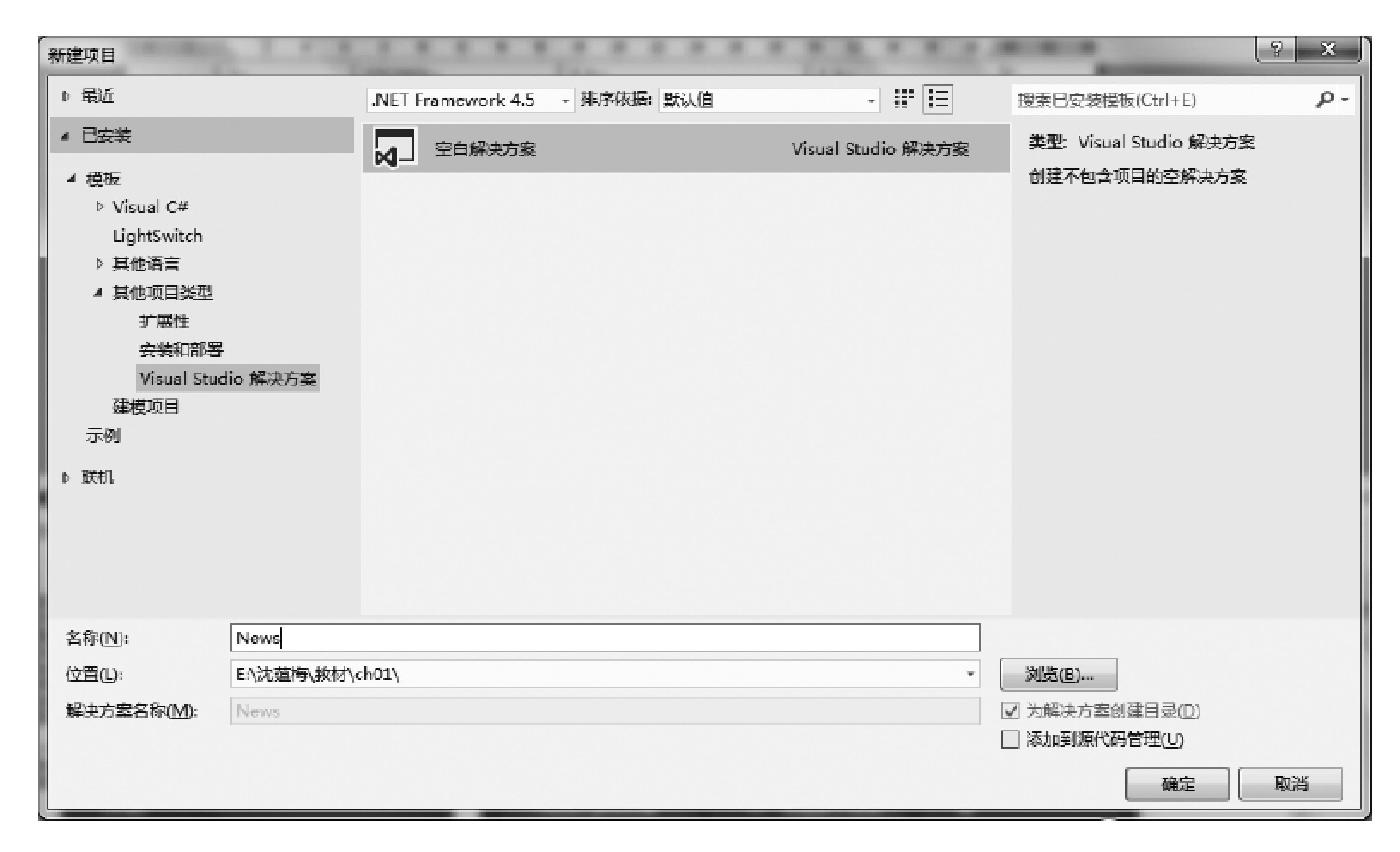


图 1.1 【新建项目】对话框



图 1.2 新建项目

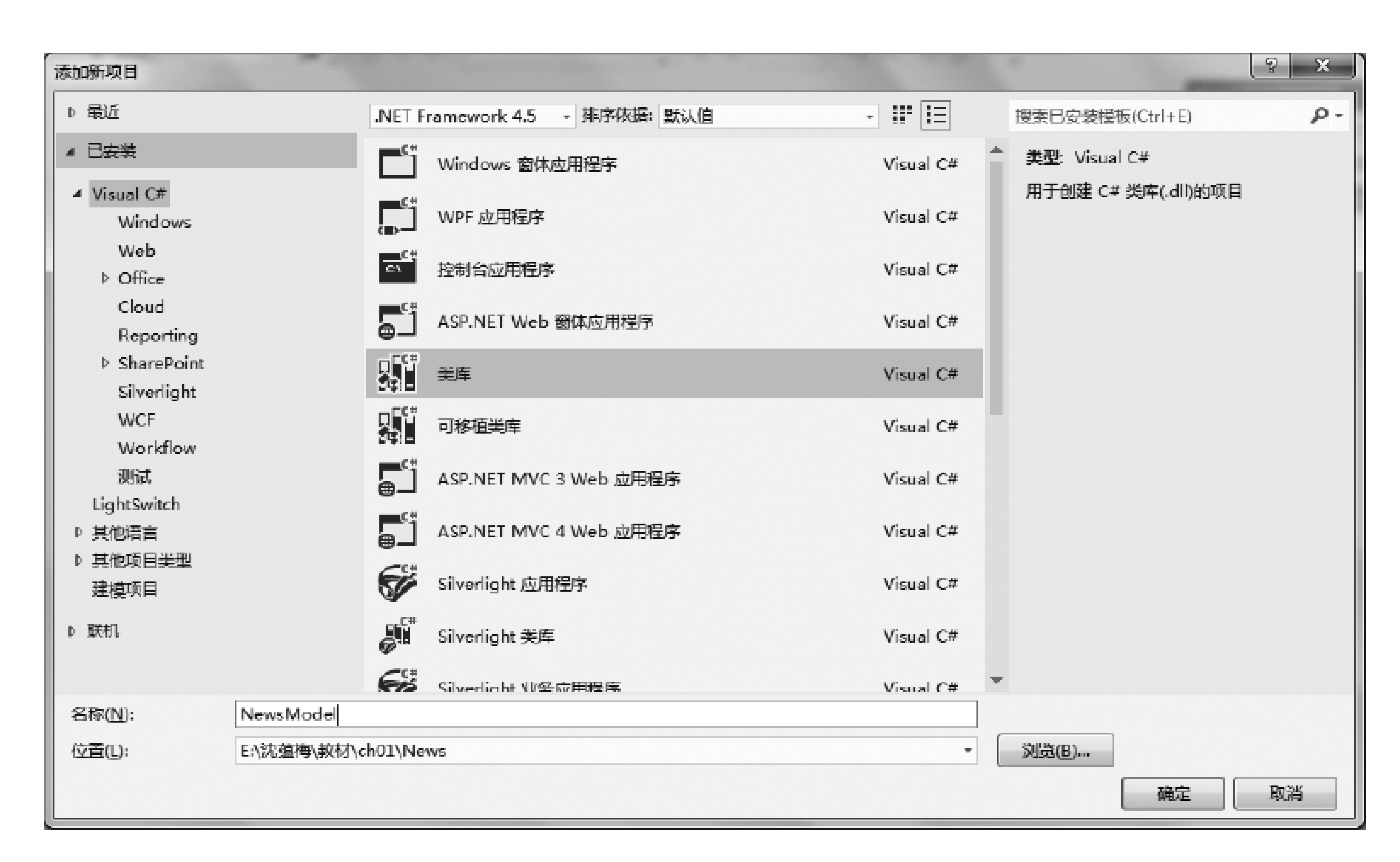


图 1.3 【添加新项目】对话框

【步骤 3】添加数据访问层。

【步骤 4】添加业务逻辑层。

数据访问层和业务逻辑层的添加与实体层是类似的，数据访问层的项目命名为解决方案名＋DAL，即“NewsDAL”，业务逻辑层的项目命名为解决方案名＋BLL，即“NewsBLL”，添加完数据访问层和业务逻辑层后，解决方案资源管理器如图 1.4 所示。

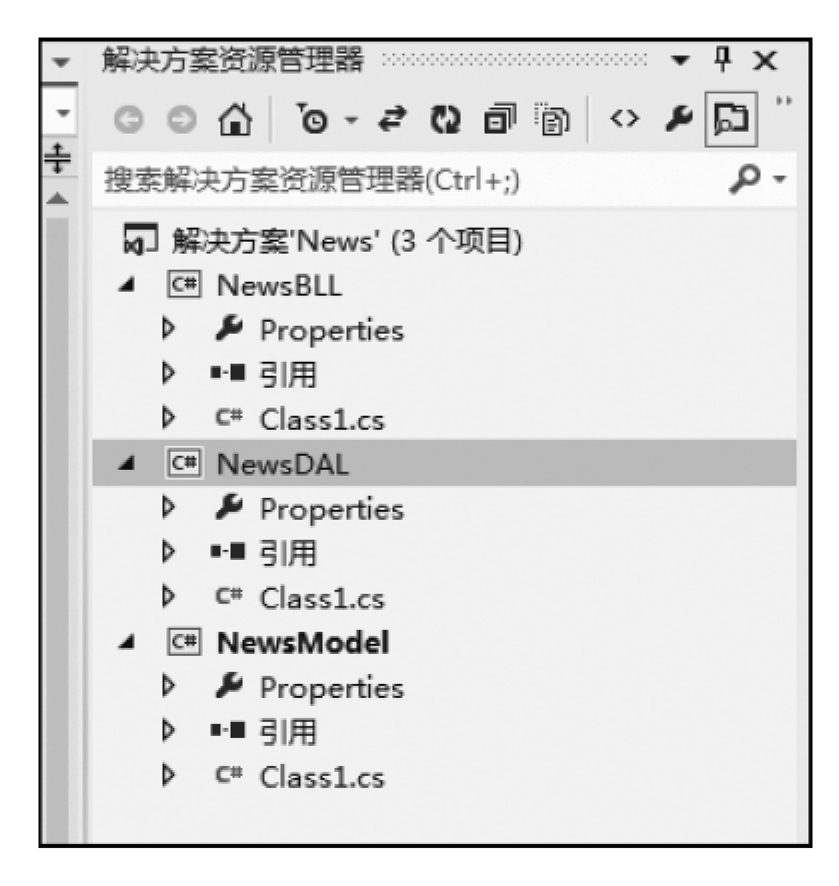


图 1.4　解决方案资源管理器

【步骤 5】添加表示层。

右击解决方案，选择“添加”→“新建网站”，注意表示层的 Web 位置要选择解决方案所在文件夹，这里我们将表示层的 Web 位置设定为“E:\教材\ch01\News\Web”，其中 Web 为表示层的命名，如图 1.5 所示。



图 1.5　【添加新网站】对话框

【步骤 6】搭建后的架构在解决方案资源管理器中的效果如图 1.6 所示，目录结构如图 1.7 所示。




图 1.6　解决方案资源管理器



图 1.7　目录结构图

1.2.3　系统逻辑设计

本部分内容选择第 10 章使用的案例—新闻发布系统，以便大家对案例的整体结构有所了解。系统的功能模块如图 1.8 所示。

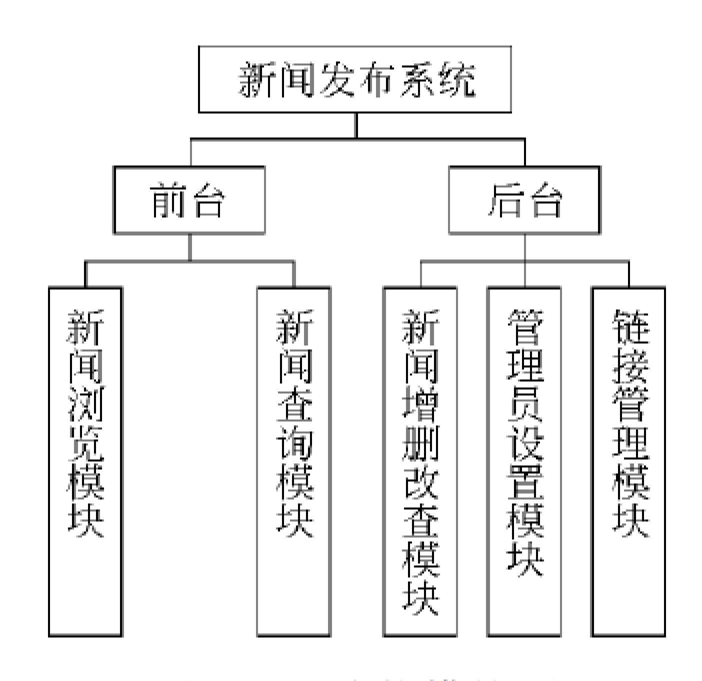


图 1.8　功能模块图

1.2.4　数据库创建方法

本系统采用 SQL Server 2008 数据库，包含若干数据表，具体如图 1.9～图 1.12 所示。

PC201011030... dbo.tbLink

列名	数据类型	允许 Null 值
ID	int	☐
picPath	varchar(50)	☑
linkName	varchar(50)	☑
linkAddress	varchar(50)	☑
addDate	datetime	☑
		☐

图 1.9　链接表

PC201011030... dbo.tbUser

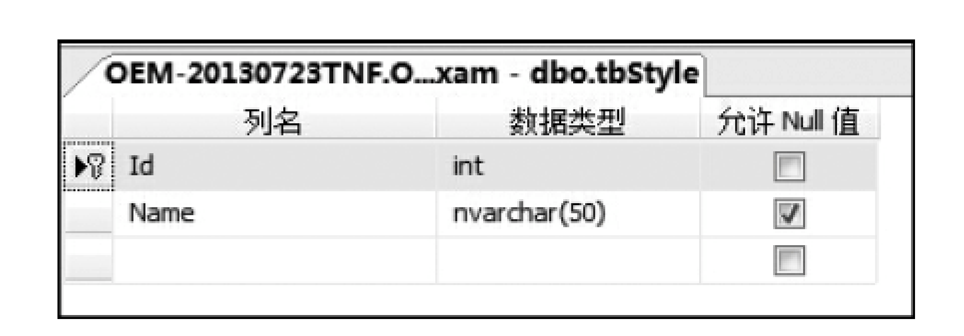
列名	数据类型	允许 Null 值
ID	int	☐
Name	varchar(20)	☐
PassWord	varchar(50)	☐
addDate	datetime	☐

图 1.10　管理员表

PC201011030... dbo.tbNews

列名	数据类型	允许 Null 值
ID	int	☐
Title	varchar(50)	☑
[Content]	text	☑
StyleId	int	☑
IssueDate	smalldatetime	☑
imageid	nvarchar(50)	☑
		☐

图 1.11　新闻表


OEM-20130723TNF.O...xam - dbo.tbStyle

列名	数据类型	允许 Null 值
Id	int	☐
Name	nvarchar(50)	☑
		☐

图 1.12　类别表

1.3 项目训练

通过对以上内容的学习，了解了搭建系统架构的一般步骤，现在回到项目导入的任务中来。

【步骤1】按照上面所述内容，搭建系统架构，如图1.13所示。

【步骤2】通过分析，最终确定系统功能模块图如图1.14所示。



图1.13 系统架构图

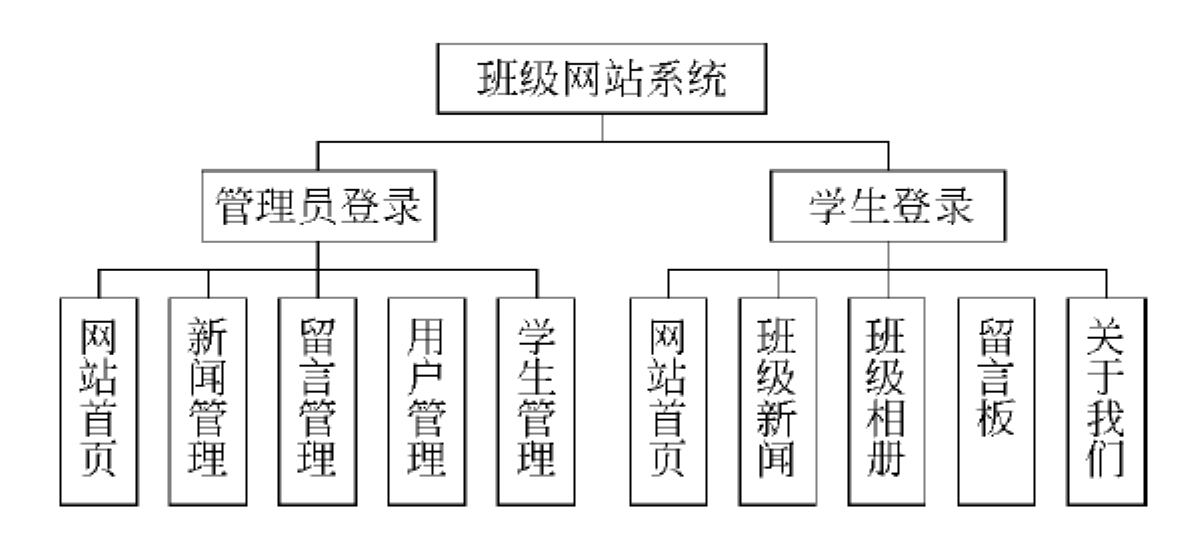


图1.14 功能模块图

【步骤3】班级网站数据表如图1.15～图1.19所示。

OEM-20130723TNF....sDB - dbo.Photos

列名	数据类型	允许 Null 值
Id	int	☐
PhotoName	nvarchar(MAX)	☐

图1.15 班级照片表

OEM-20130723TNF....assDB - dbo.News

列名	数据类型	允许 Null 值
Id	int	☐
Title	nvarchar(MAX)	☑
[Content]	nvarchar(MAX)	☑
AddTime	datetime	☑

图1.16 班级新闻表

OEM-20130723TNF....B - dbo.Students

列名	数据类型	允许 Null 值
Id	int	☐
StudentNo	nvarchar(50)	☑
StudentName	nvarchar(50)	☑
Sex	nvarchar(50)	☑
Phone	nvarchar(50)	☑
Address	nvarchar(MAX)	☑
Email	nvarchar(50)	☑
IdentityCard	nvarchar(50)	☑
Grades	nvarchar(50)	☑

图1.17 学生信息表

OEM-20130723TNF....- dbo.GuestBooks

列名	数据类型	允许 Null 值
Id	int	☐
Name	nvarchar(10)	☑
Face	nvarchar(50)	☑
[Content]	nvarchar(MAX)	☑
AddTime	datetime	☑

图1.18 留言板表

OEM-20130723TNF....B - dbo.UserInfos

列名	数据类型	允许 Null 值
UserId	int	☐
UserName	nvarchar(50)	☑
UserPassword	nvarchar(50)	☑
UserType	nvarchar(50)	☑

图1.19 用户信息表

1.4　平行项目训练

1. 训练内容

大三学生小刘想为母校设计一款在线考试系统，能够实现在线考试。

2. 训练目的

(1) 进一步训练和巩固学生对 VS2012 搭建系统架构的步骤的理解；

(2) 使学生对功能模块图、数据库创建有比较深刻的认识。

3. 训练过程

【步骤 1】搭建系统架构，如图 1.20 所示。

【步骤 2】绘制功能模块图，如图 1.21 所示。

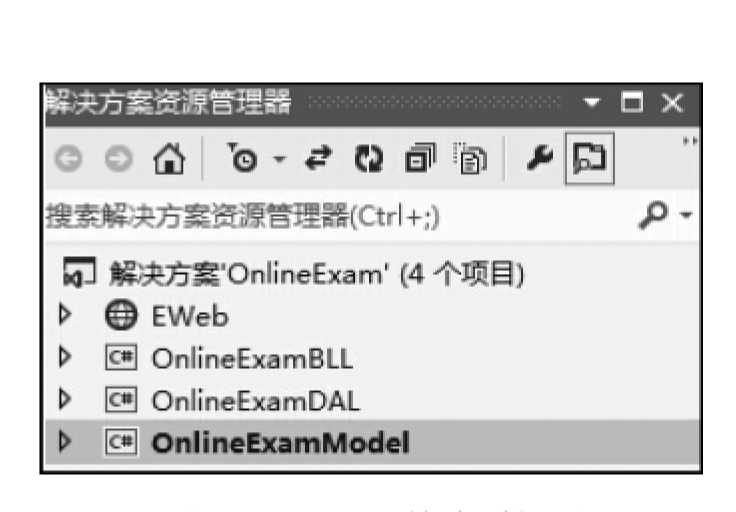


图 1.20　系统架构图

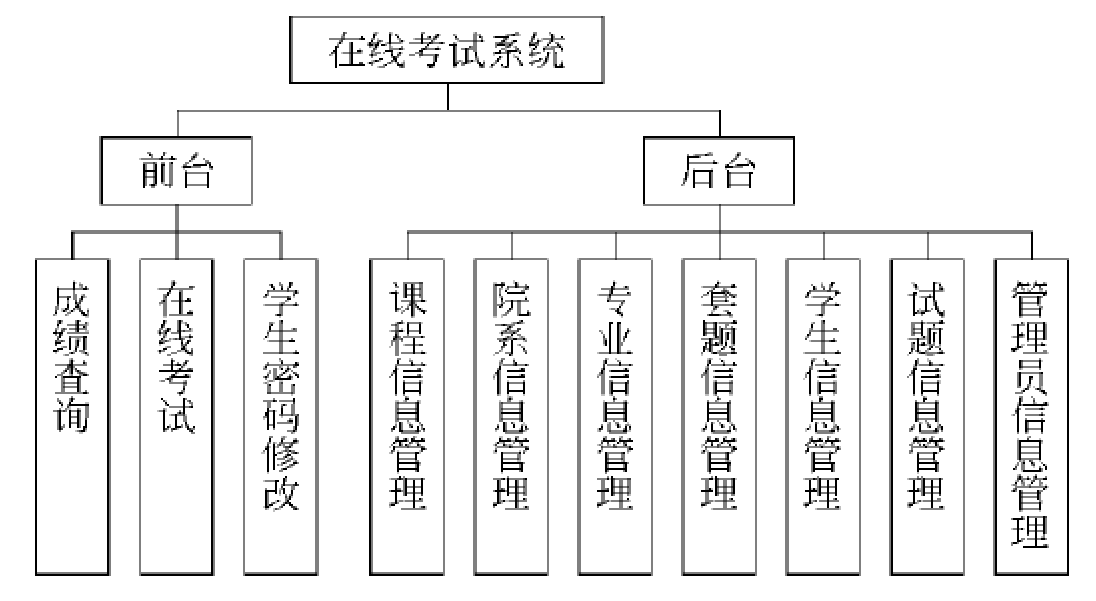


图 1.21　功能模块图

【步骤 3】创建数据库，数据表如图 1.22～图 1.28 所示。

OEM-20130723TNF....Exam - dbo.Admin

列名	数据类型	允许 Null 值
ID	bigint	☐
Name	varchar(50)	☐
Pwd	varchar(50)	☐
JoinTime	datetime	☑
		☐

图 1.22　管理员表



OEM-20130723TNF....Exam - dbo.Lesson

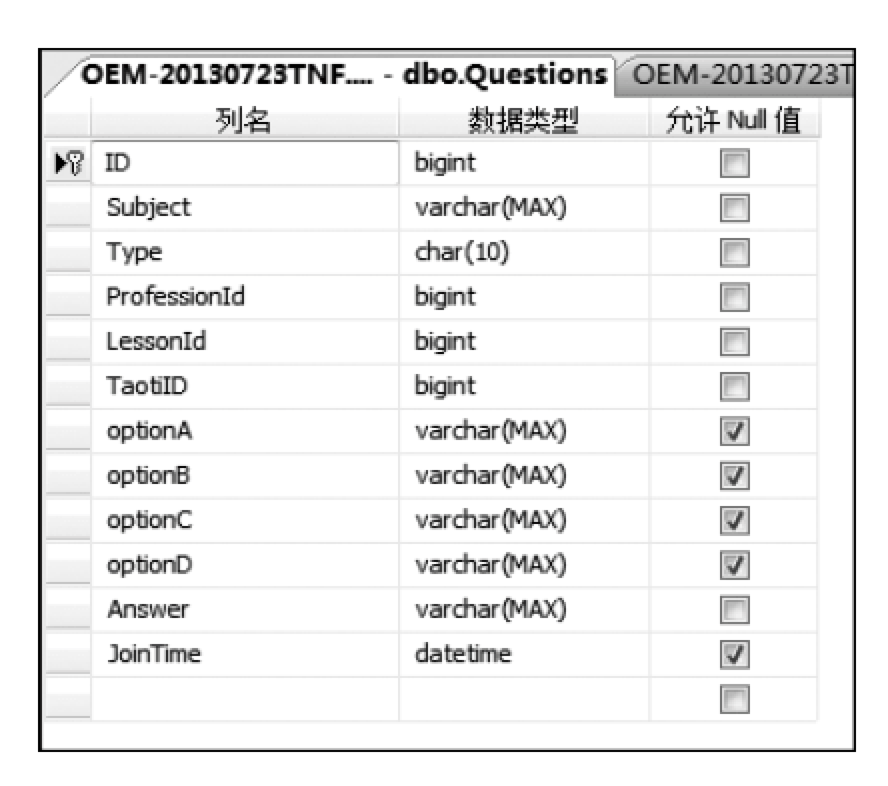
列名	数据类型	允许 Null 值
ID	bigint	☐
Name	varchar(50)	☐
ProfessionId	bigint	☐
JoinTime	datetime	☑
		☐

图 1.23　课程表

OEM-20130723TNF.... - dbo.Profession

列名	数据类型	允许 Null 值
ID	bigint	☐
Name	varchar(50)	☐
JoinTime	datetime	☑
		☐

图 1.24　专业表



OEM-20130723TNF.... - dbo.Questions　OEM-20130723T

列名	数据类型	允许 Null 值
ID	bigint	☐
Subject	varchar(MAX)	☐
Type	char(10)	☐
ProfessionId	bigint	☐
LessonId	bigint	☐
TaotiID	bigint	☐
optionA	varchar(MAX)	☑
optionB	varchar(MAX)	☑
optionC	varchar(MAX)	☑
optionD	varchar(MAX)	☑
Answer	varchar(MAX)	☐
JoinTime	datetime	☑
		☐

图 1.25　试题表

OEM-20130723TNF.... - dbo.StuRestult

列名	数据类型	允许 Null 值
ID	bigint	☐
StuId	varchar(16)	☐
LessonId	bigint	☐
TaoTiId	bigint	☐
ResSingle	int	☑
ResMore	int	☑
ResTotal	int	☑
JoinTime	datetime	☐
		☐

图 1.26　成绩表

OEM-20130723TNF....am - dbo.Student

列名	数据类型	允许 Null 值
ID	varchar(16)	☐
Name	varchar(20)	☐
Pwd	varchar(50)	☐
Sex	char(2)	☑
Question	varchar(50)	☐
Answer	varchar(50)	☐
ProfessionId	bigint	☐
cardID	varchar(20)	☐
Jointime	nchar(10)	☑
		☐

图 1.27　学生表

OEM-20130723TNF....Exam - dbo.TaoTi

列名	数据类型	允许 Null 值
ID	bigint	☐
Name	varchar(50)	☐
lessonID	bigint	☐
JoinTime	datetime	☑
		☐

图 1.28　套题表

1.5　总　　结

本章通过简单项目示例，介绍了 ASP. NET 的发展历史以及 ASP. NET、. NET Framework、Visual Studio 版本的演进历史，并详细介绍了搭建系统框架的步骤，以满足项目实现的需求，通过贯穿项目“新闻发布系统”和平行项目“在线考试系统”系统地学习了框架的搭建、功能模块的绘制以及数据库的创建，增强了学生学习 ASP. NET 程序设计的信心和情趣。

1.6　习　　题

1. Visual Studio 2012 新增了哪些功能特性？
2. 如何搭建系统框架？
3. 画出学生管理系统的功能模块图。

第 2 章　实体设计与多层依赖

本章要点：

- 创建多层依赖关系
- 创建实体类

技能目标：

- 会创建多层之间的依赖关系
- 会根据数据库表创建实体类

2.1　项目导入

【项目场景】

在第 1 章中，我们搭建了系统架构，在解决方案中创建了四个独立项目，接下来我们就要创建四个独立项目之间的依赖关系以及实体类，如图 2.1 所示。

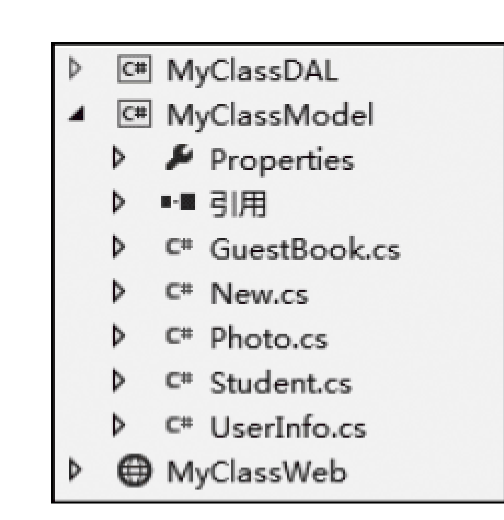


图 2.1　创建实体类及依赖关系

【问题引导】

(1) 如何创建依赖关系。

(2) 如何创建实体类。

2.2　技术与知识准备

2.2.1　根据数据库表编写实体

实体层包含所有与数据库中的表相对应的实体类。可以说，实体层提供了一个标准和规范，三层之间的数据传递就是通过传输实体对象来达到目的。

实体层中的实体类一般和所对应的表名一致。实体类比较简单，根据数据库中的字段编写对应的变量和属性即可。除了构造函数以外，实体类一般没有其他方法。

【示例 2.1】　现在我们编写第 1 章中新闻表的实体类，鉴于新闻表有一个外键，所以我们需要创建两个对应的实体类。如图 2.2 所示。

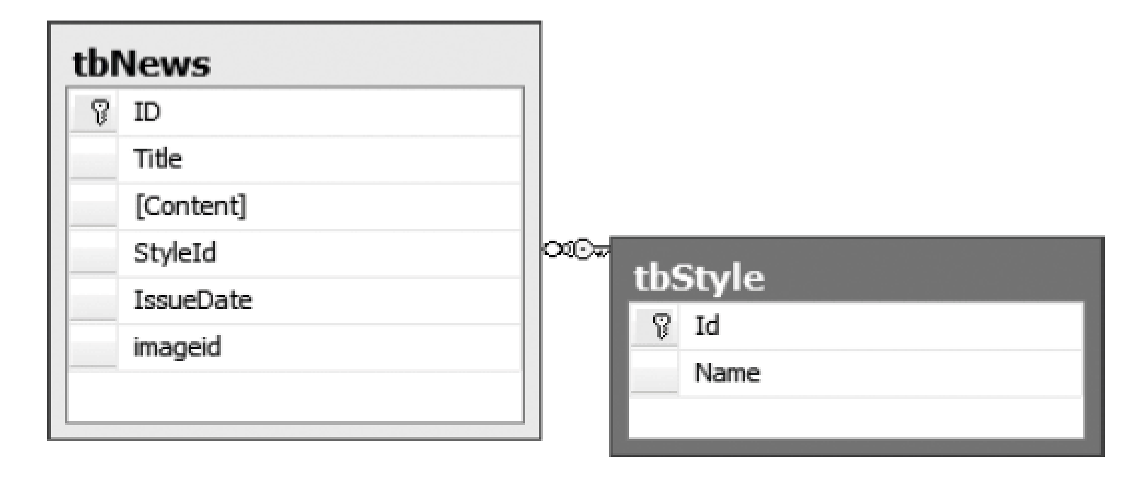


图 2.2　新闻表及其外键

实体类的编写比较简单，限于篇幅，我们只给出 New 类的完整代码。

```
namespace NewsModels
{
   public class tbNew
    {
         private int iD;
         public int ID
         {
              get { return iD; }
              set { iD=value; }
         }
         private string title;
         public string Title
         {
              get { return title; }
              set { title=value; }
         }
         private string content;
         public string Content
         {
              get { return content; }
              set { content=value; }
         }
         private tbStyl style;
         public tbStyl Style
         {
              get { return style; }
              set { style=value; }
         }
         private DateTime issueDate;
         public DateTime IssueDate
         {
              get { return issueDate; }
              set { issueDate=value; }
```

```
            }
            private string imageid;
            public string Imageid
            {
                get { return imageid; }
                set { imageid=value; }
            }
        }
    }
    namespace NewsModels
    {
        public class tbStyl
        {
            public int Id { set; get; }
            public string Name { set; get; }
        }
    }
```

按照此方法，依次创建新闻发布系统的各个实体类，如图 2.3 所示，实体类和数据库表数据类型对应关系如表 2.1 所示。

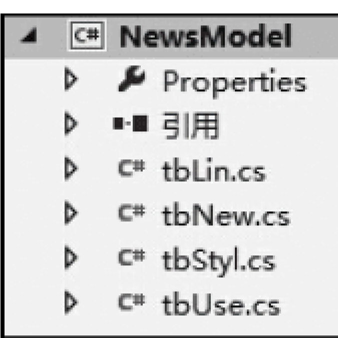


图 2.3　新闻发布系统的实体类

表 2.1　实体类和数据库表数据类型对应关系

实体类属性类型	数据库类型	实体类属性类型	数据库类型
string	char，nchar，varchar，nvarchar	float	float
int	int ，smallint	bool	byte
DateTime	datetime	decimal	decimal，money

注意：外键的处理，采用的是使用外键对象，这种方式的好处是：它可以依据外键类直接访问外键的其他属性，定义时如上题 public tbStyl Style，tbStyl 是类名，Style 是属性名，使用时可以这样访问新闻类别：tbNew tbn＝new tbNew()；tbn. Style. Name。

说明：

(1) 如果表名以“s”结尾，例如新闻表中 tbNews，但实体类一般会以单数形式 tbNew 表示。为避免混淆，表名与类名尽量不一致。

(2) 实体类中的属性名可与数据表的字段名一致。

2.2.2　实现各层之间的依赖

创建完系统架构后，各层之间还是相互独立的，必须建立起如图 2.4 所示的各层之间的依赖关系。

【示例 2.2】 为新闻发布系统添加各层之间的依赖关系。

【步骤 1】表示层添加引用关系。

在解决方案资源管理器中，右击 MyWeb，在弹出的快捷菜单中选择【添加引用】选项，弹出添加引用对话框，如图 2.5 所示，选择项目选项卡，依次选择 NewsBLL、NewsModel，单击“确定”按钮，完成表示层的添加引用关系。

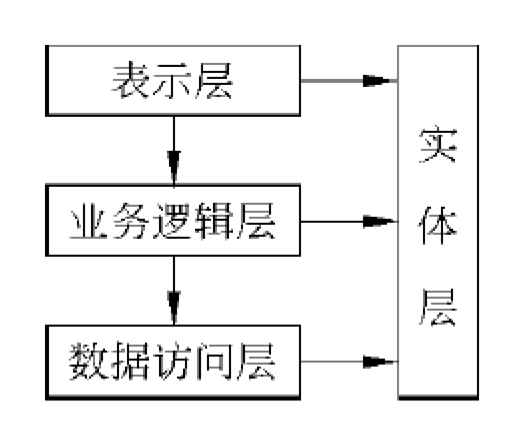


图 2.4 各层之间依赖关系

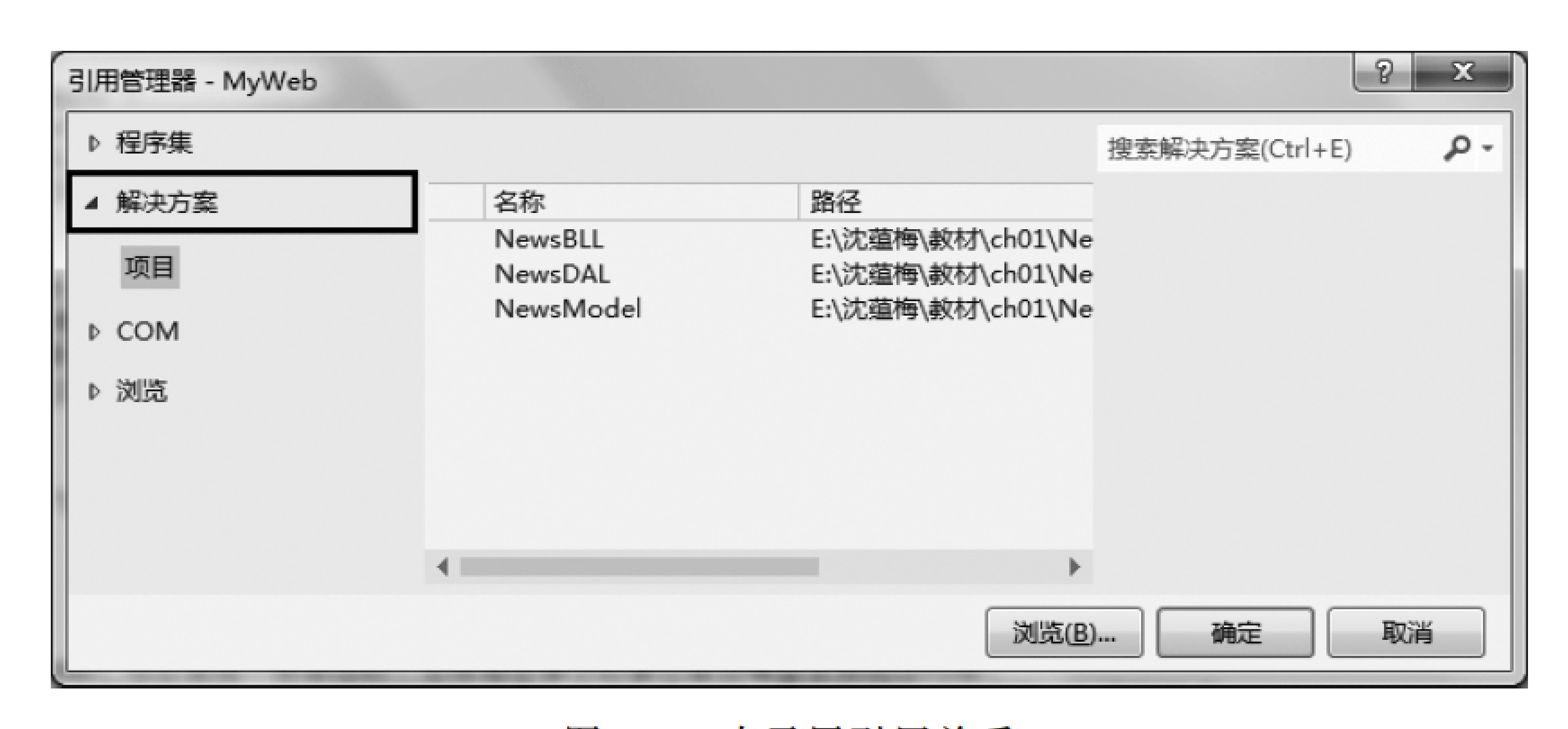


图 2.5 表示层引用关系

【步骤 2】业务逻辑层添加引用关系。

同理，右击 NewsBLL，依次添加对 NewsDAL、NewsModel 引用关系，如图 2.6 所示。

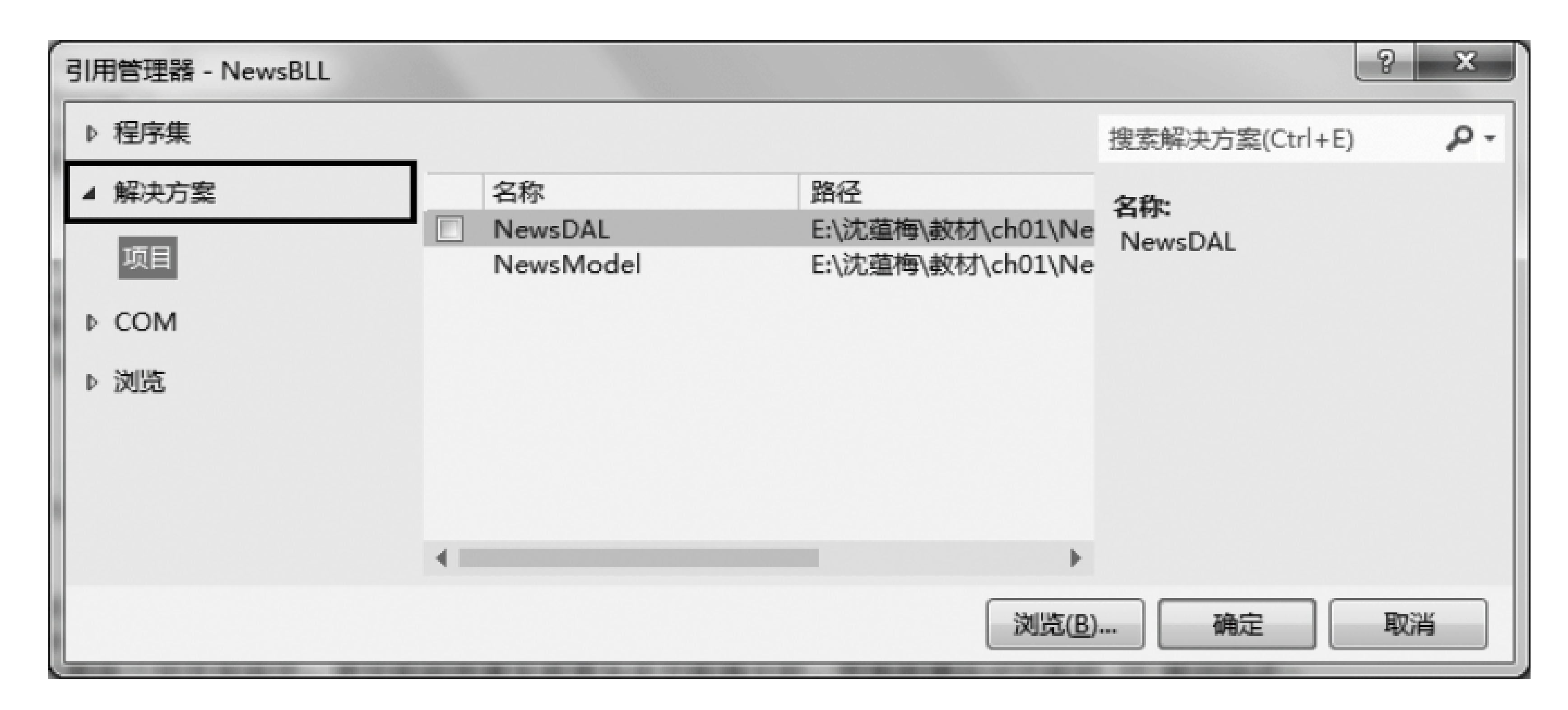


图 2.6 业务逻辑层引用关系

【步骤 3】数据访问层添加引用关系。

同理，右击 NewsDAL，添加对 NewsModel 的引用关系，如图 2.7 所示。

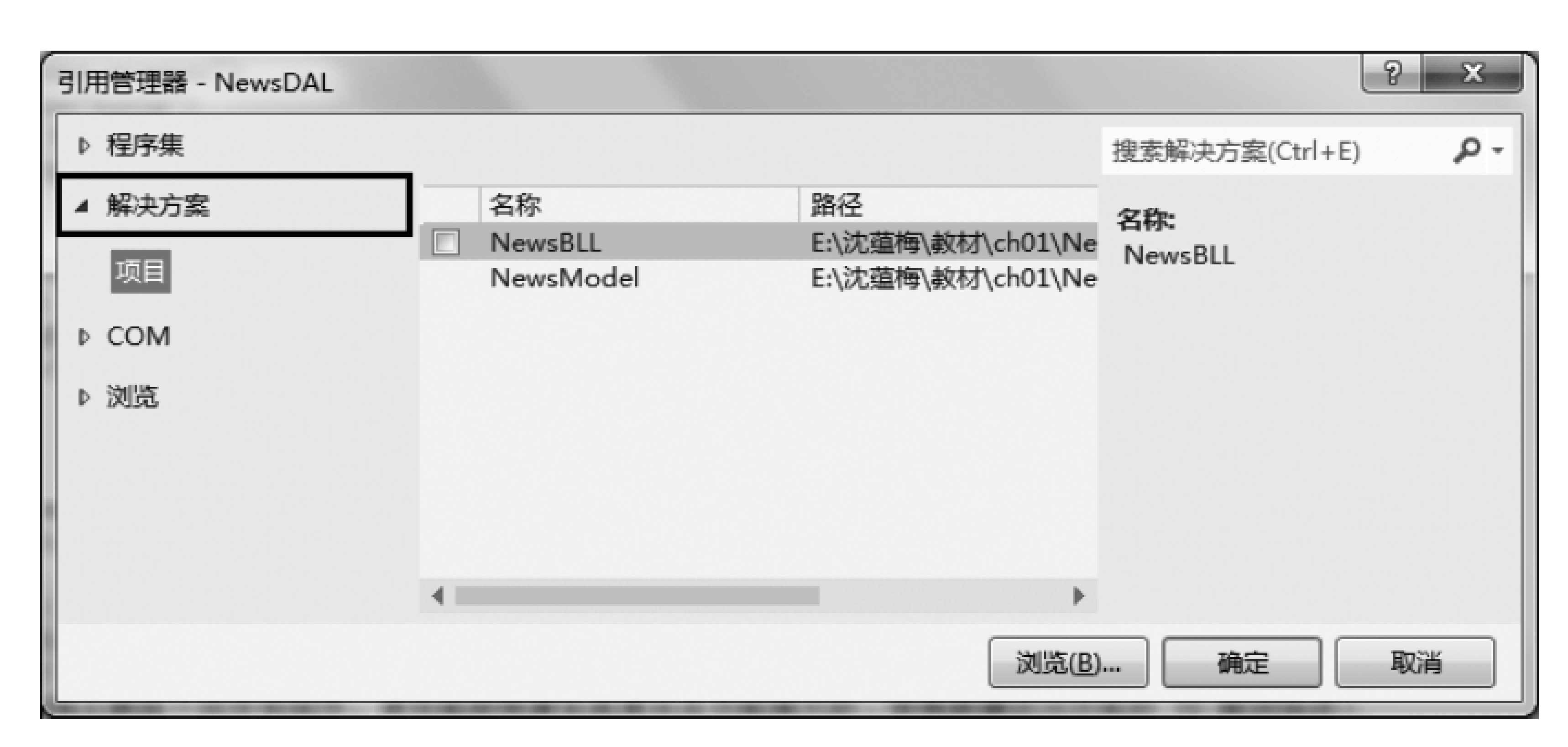


图 2.7　数据访问层引用关系

2.3　项 目 训 练

通过对以上内容的学习，了解了创建多层之间依赖关系的一般步骤，同时了解了创建实体类的方法，现在回到项目导入的任务中来。

1. 按照上面所述内容，创建多层之间依赖关系。

【步骤 1】创建表示层的依赖关系，依次添加对 MyClassBLL、MyClassModel 层的引用，如图 2.8 所示。

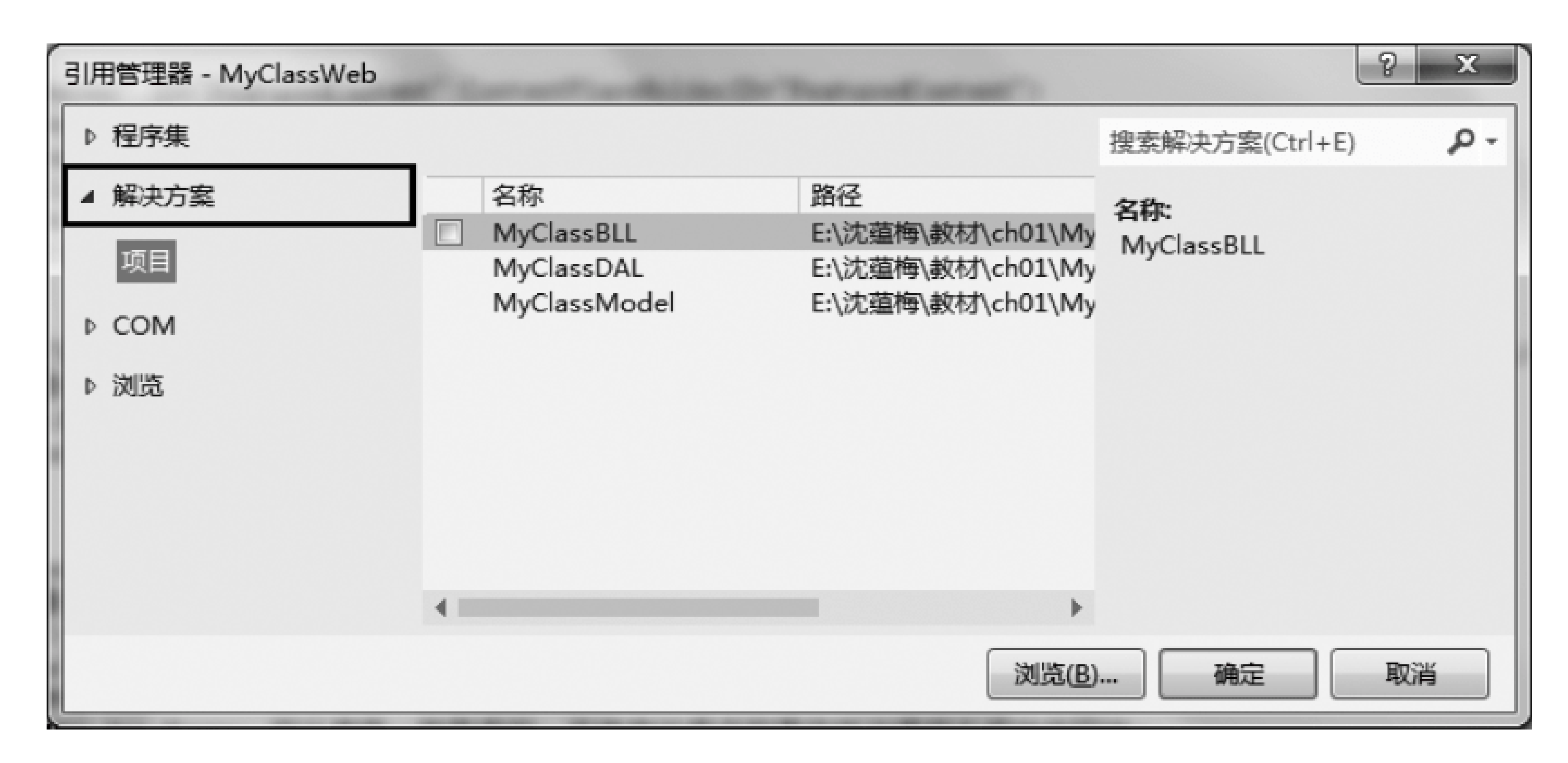


图 2.8　表示层引用关系

【步骤 2】创建业务逻辑层的依赖关系，依次添加对 MyClassDAL、MyClassModel 层的引用，如图 2.9 所示。

【步骤 3】创建数据访问层的依赖关系，添加对 MyClassModel 层的引用，如图 2.10 所示。

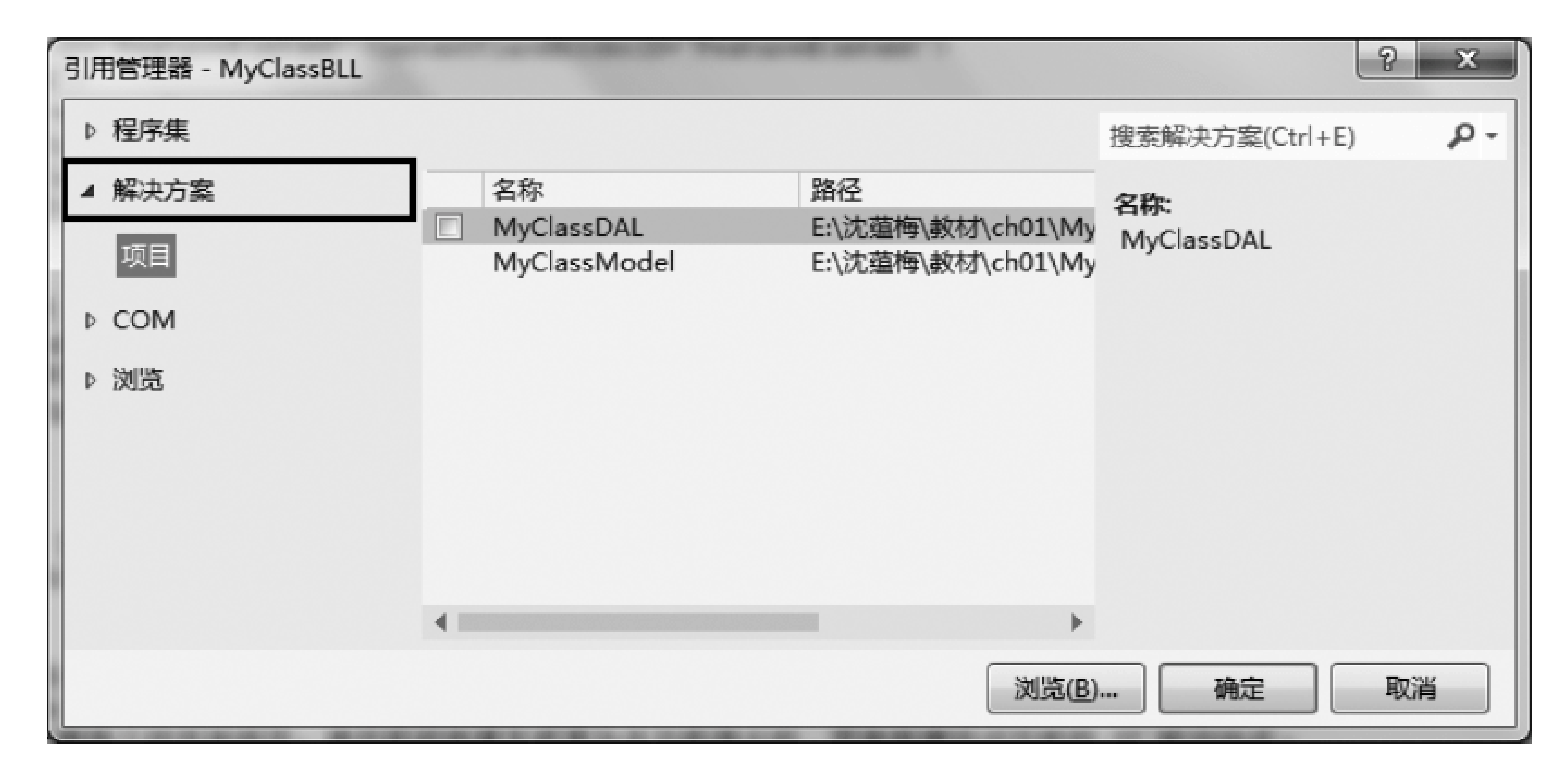


图 2.9　业务逻辑层引用关系

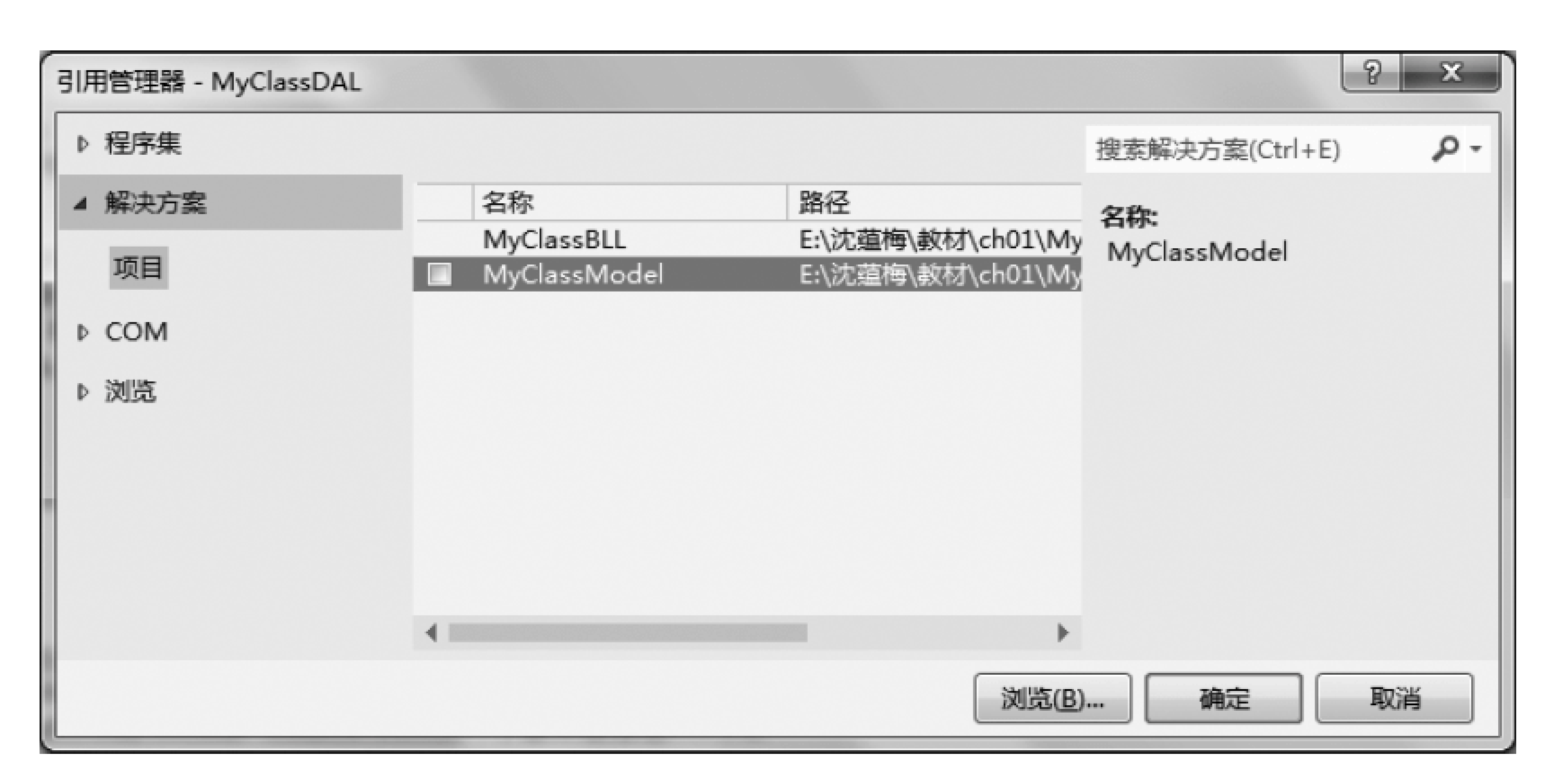


图 2.10　数据访问层引用关系

2. 根据数据库表创建各实体类(篇幅有限,只创建留言表的实体类,其他各类参考留言表的实体类)。

```
public class GuestBook
{
    private int id;
    public int Id
    {
        get { return id; }
        set { id=value; }
    }
    private string name;
    public string Name
    {
        get { return name; }
        set { name=value; }
```

```
        }
        private string face;
        public string Face
        {
            get { return face; }
            set { face=value; }
        }
        private string content;
        public string Content
        {
            get { return content; }
            set { content=value; }
        }
        private DateTime addtime;
        public DateTime Addtime
        {
            get { return addtime; }
            set { addtime=value; }
        }
    }
```

2.4　平行项目训练

1. 训练内容

根据数据库表创建实体层的实体类,并能够创建各层之间的依赖关系。

2. 训练目的

(1) 进一步训练和巩固学生对实体层、实体类概念及其创建方法的理解;

(2) 使学生对创建多层依赖关系有比较深刻的认识。

3. 训练过程

【步骤 1】创建多层之间的依赖关系。

(1) 创建表示层的依赖关系,依次添加对 OnlineExamBLL、OnlineExamModel 层的引用,如图 2.11 所示。

(2) 创建业务逻辑层的依赖关系,依次添加对 OnlineExamDAL、OnlineExamModel 层的引用,如图 2.12 所示。

(3) 创建数据访问层的依赖关系,添加对 OnlineExamModel 层的引用,如图 2.13 所示。

【步骤 2】创建实体类(篇幅有限,只创建学生表的实体类,其他各类参考学生表的实体类)。



图 2.11　表示层引用关系

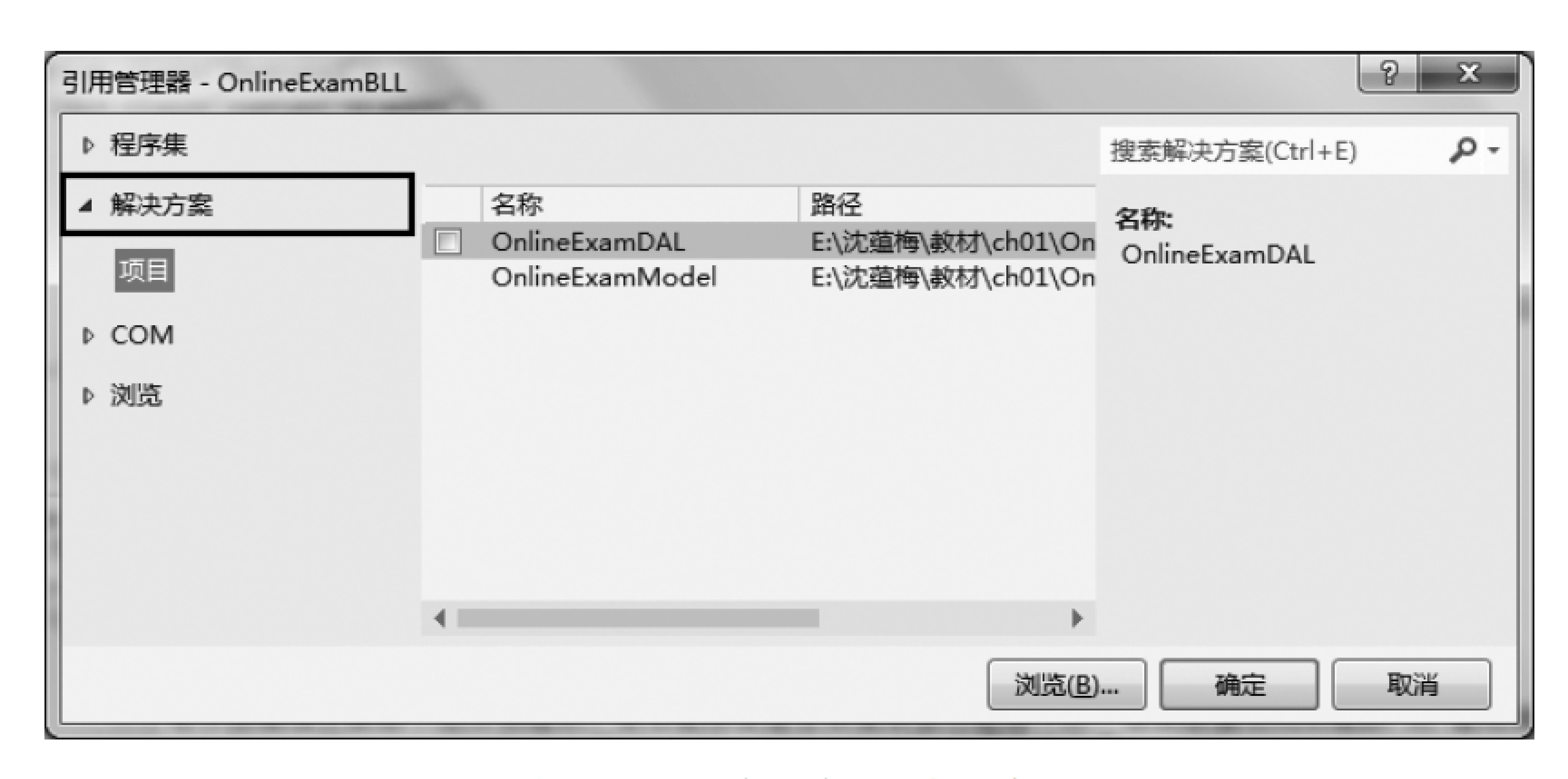


图 2.12　业务逻辑层引用关系

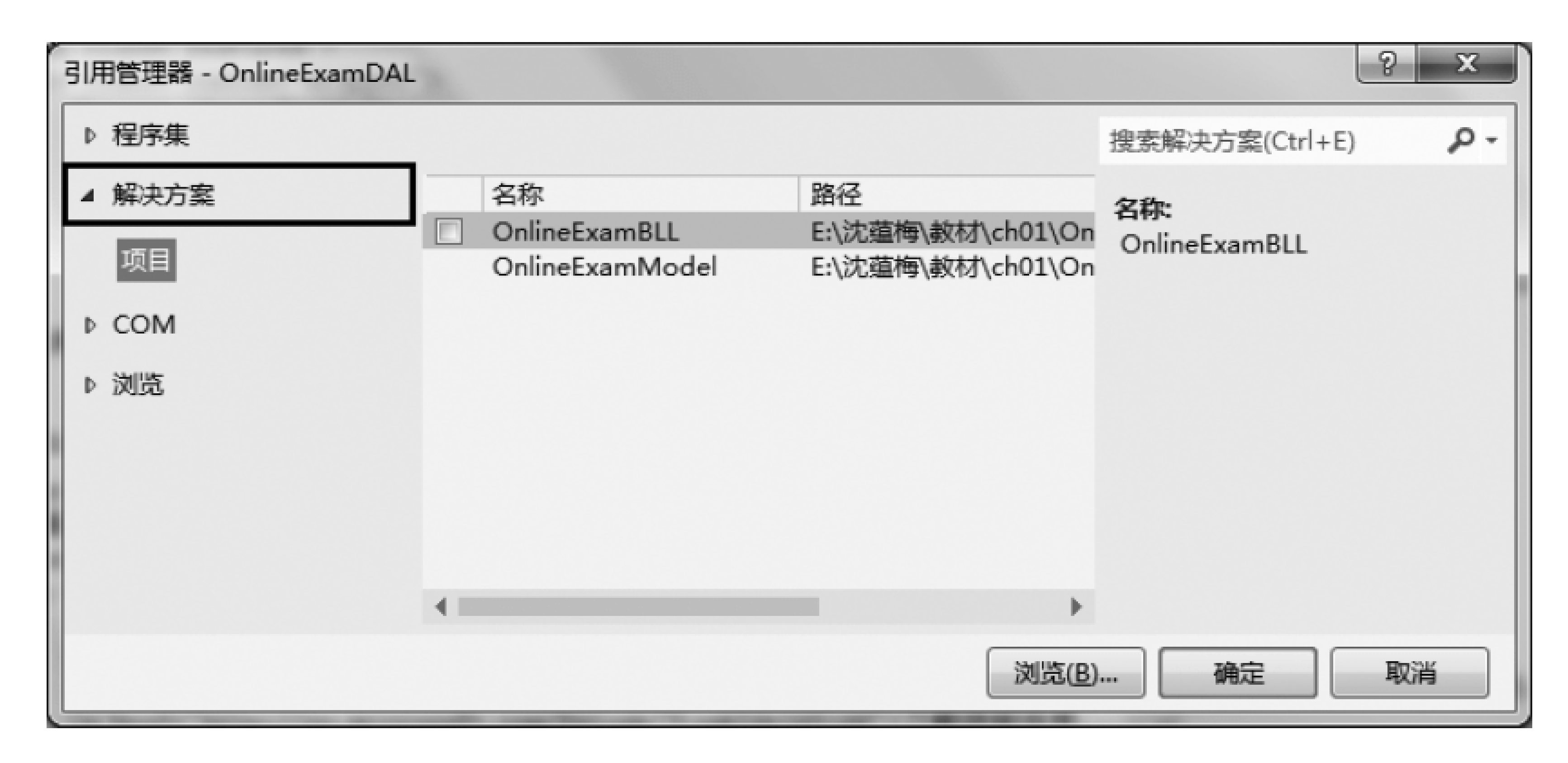


图 2.13　数据访问层引用关系

```
public class Stud
    {
        private string iD;
        public string ID
        {
            get { return iD; }
            set { iD=value; }
        }
        private string name;
        public string Name
        {
            get { return name; }
            set { name=value; }
        }
        private string pwd;
        public string Pwd
        {
            get { return pwd; }
            set { pwd=value; }
        }
        private string sex;
        public string Sex
        {
            get { return sex; }
            set { sex=value; }
        }
        private string question;
        public string Question
        {
            get { return question; }
            set { question=value; }
        }
        private string answer;
        public string Answer
        {
            get { return answer; }
            set { answer=value; }
        }
        private Prof profession;
        public Prof Profession
        {
            get { return profession; }
            set { profession=value; }
        }
```

```
        private string cardID;
        public string CardID
        {
            get { return cardID; }
            set { cardID=value; }
        }
        private string jointime;
        public string Jointime
        {
            get { return jointime; }
            set { jointime=value; }
        }
    }
public  class Prof
    {
        private int iD;
        public int ID
        {
            get { return iD; }
            set { iD=value; }
        }
        private string name;
        public string Name
        {
            get { return name; }
            set { name=value; }
        }
        private DateTime joinTime;
        public DateTime JoinTime
        {
            get { return joinTime; }
            set { joinTime=value; }
        }
    }
```

注意：学生表的专业号是外键，要使用外键对象。

2.5 总 结

本章通过简单项目案例，介绍了如何创建多层之间的依赖关系，如何根据数据表创建相应的实体类。通过项目训练“个人网站”及平行项目训练“在线考试系统”，整体训练了如何创建四个独立项目之间的依赖关系，如何创建实体类以及外键的处理方法。

2.6　习　　题

1. 什么是实体类？实体类有什么作用？
2. 在实体类中如何处理外键？
3. 完成新闻发布系统实体类的创建。
4. 完成在线考试系统实体类的创建。

第 3 章　数据库访问与实现

本章要点：

- ADO.NET 的功能与组成
- 创建数据库操作类

技能目标：

- 会使用 Connection 对象连接到数据库
- 会创建访问数据库的操作类
- 会编写简单的数据库查询语句

3.1　项目导入

【项目场景】

苏州健雄学院学生处想要开发一个学生查询系统，根据学生学号，查询学生详细信息，请你为学生处开发一个系统，实现信息查询功能，如图 3.1 所示。

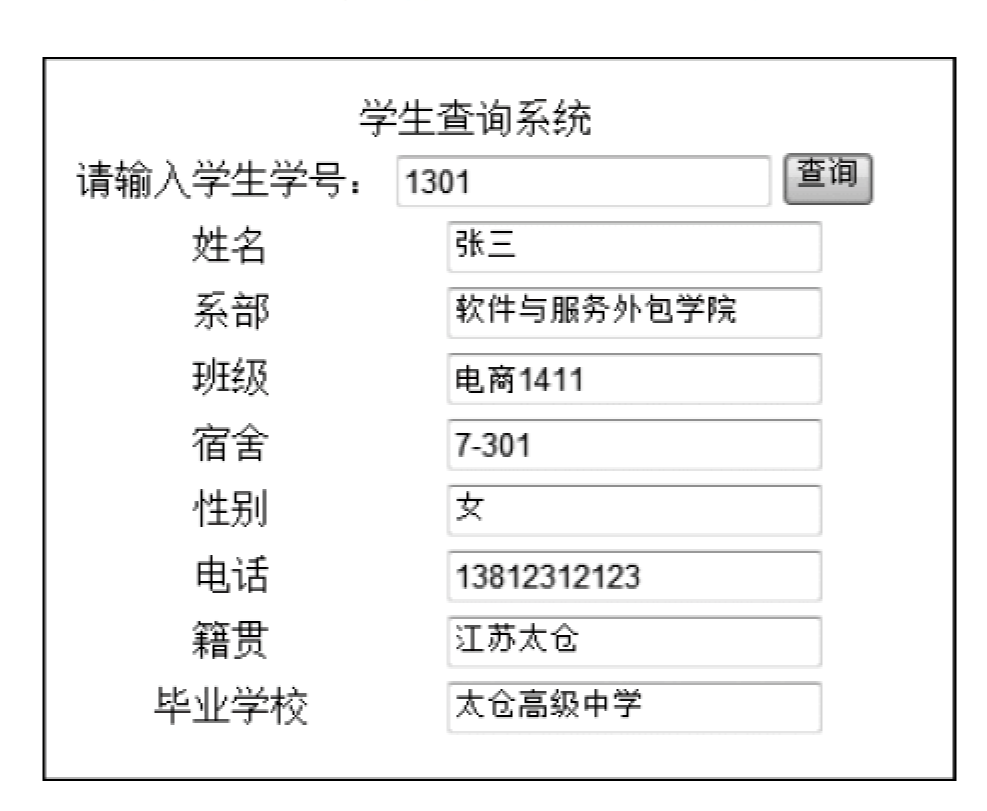


图 3.1　学生查询系统

【问题引导】

(1) 如何进行页面布局。
(2) 如何实现数据库数据的访问。

（3）如何实现数据的查询功能以满足信息管理的需要。

3.2　技术与知识准备

3.2.1　访问数据方法和编码

ADO. NET 是. NET Framework 中用于数据访问的组件，它由 Microsoft ActiveX Data Object（ADO）改进而来，是一组用于和数据源进行交互的面向对象类库。通常来说，数据源是数据库，但也可以是 Excel 表格或者 XML 文件等。ADO. NET 允许和不同类型的数据源以及数据库进行交互。微软公司认为，ADO. NET 是对早期 ADO 技术的"革命性改进"。应该说，它确实是一种非常优秀的数据访问技术，对于使用. NET Framework 进行软件开发的程序员来说，它是必须掌握的技术之一。ADO. NET 提供与数据源进行交互的相关的公共方法，但是对于不同的数据源采用一组不同的类库，这些类库称为 Data Providers，并且通常是以与之交互的协议和数据源的类型来命名的。ADO . NET 包含以下 5 种对象。

- Command 对象：用于与数据库交互时所执行的操作。不管是增、删、改、查操作，都要求从一个 Command 对象开始。
- Connection 对象：用于建立与数据源的连接，处理访问数据源时所需要的安全设置。
- DataReader 对象：当 Command 对象返回结果集时，需要使用 DataReader 对象来检索数据。返回值是只读的数据流。
- DataSet 对象：是数据集，它不直接绑定到数据源，可以缓存来自多个数据源的数据。
- DataAdapter 对象：一种用来充当 DataSet 对象与实际数据源之间桥梁的对象。

总之，ADO. NET 是与数据源交互的. NET 技术。有很多的 Data Provider，它将允许与不同的数据源交互。然而无论使用什么样的 Data Provider，都使用相似的对象与数据源进行交互。

3.2.2　访问类的设计与编码

在对数据库的所有操作中，数据库的打开、连接、执行 SQL 语句是很常见的操作，而且这些操作经常需要反复执行。如果每次都编写代码，势必会浪费很多精力，因此我们开发出一个执行数据库操作的数据库操作类 DBHelper. cs 类用于执行这些重复的操作。

【步骤 1】添加数据库操作类 DBHelper. cs。

在右侧解决方案资源管理器窗口右击数据访问层（DAL），在弹出的快捷菜单中选择"添加"→"类"，弹出如图 3. 2 所示对话框，在名称框中输入 DBHelper. cs，单击"确定"按钮。



图 3.2 添加数据库操作类

【步骤 2】配置 Web.config。

```
<connectionStrings>
    <add name="aa" providerName="System.Data.SqlClient" connectionString="Data Source
    =OEM-20130723TNF;Initial Catalog=dbMMChaXun;User ID=sa;Password=123456" />
  </connectionStrings>
```

【步骤 3】添加引用命名空间。

为了使用 SQL Server 的常规操作,需要在该类中添加 System.Data.SqlClient 和 System.Data 命名空间的引用。

```
using System.Data;
using System.Data.SqlClient;
```

【步骤 4】在右侧解决方案资源管理器窗口右击数据访问层(DAL),在弹出的快捷菜单中选择添加引用,弹出如图 3.3 所示对话框,在左侧程序集中选择框架,右侧 System.Configuration 前面打钩,单击确定。然后在 DBHelper.cs 类中添加引用空间:using System.Configuration。

【步骤 5】添加代码。

```
public static class DBHelper
    {    public static readonly string ConnectionString = ConfigurationManager.
         ConnectionStrings["aa"].ConnectionString;
//实现增、删、改功能
        public static int ExecuteNonQuery (string connectionString, CommandType
        commandType, string commandText, params SqlParameter[] commandPa- rameters)
        {
            using(SqlConnection connection=new SqlConnection(connectionString))
```

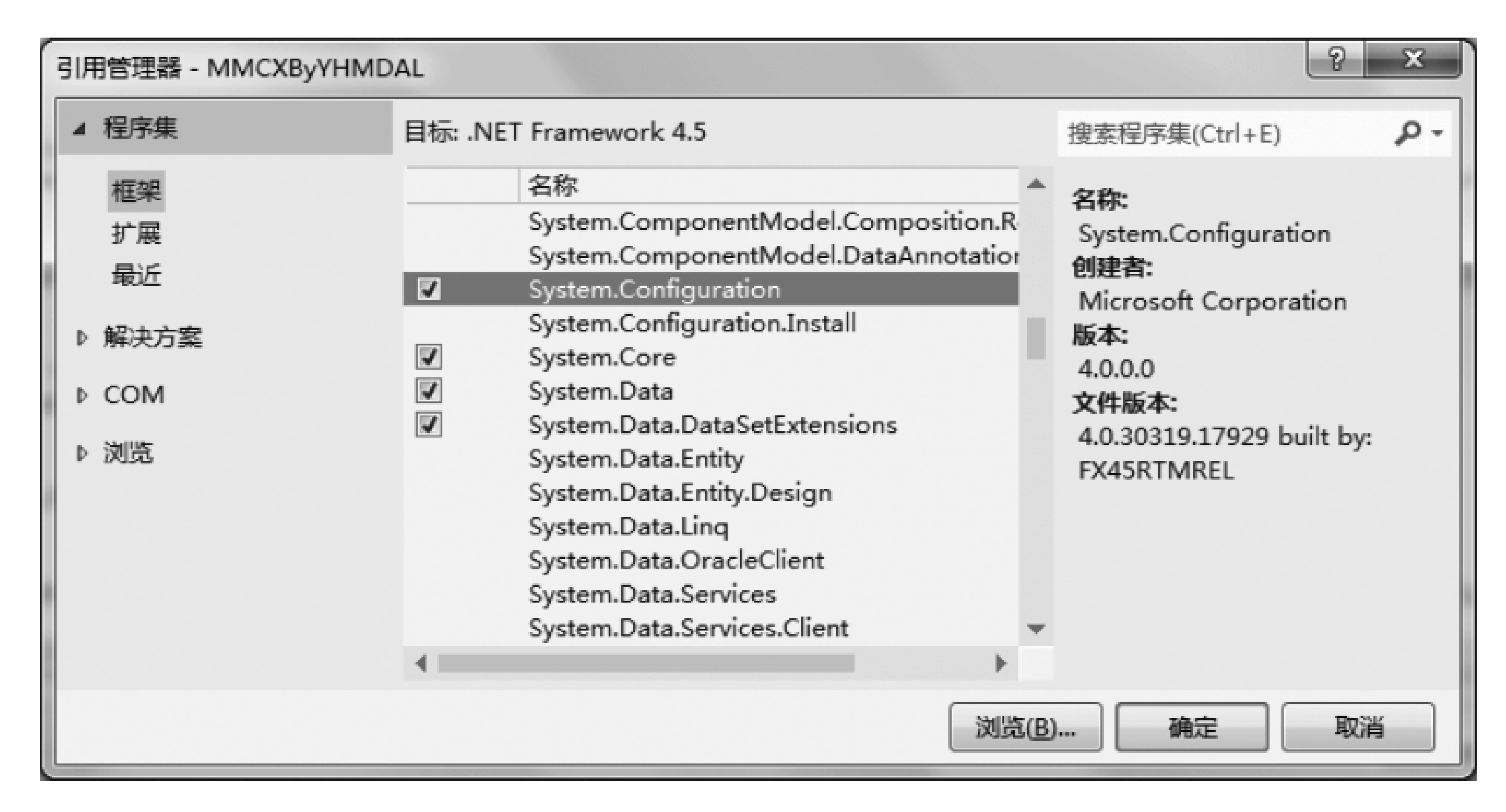


图 3.3 添加程序集

```
        {
            if(connection.State !=ConnectionState.Open)
                connection.Open();
            SqlCommand command=new SqlCommand(commandText, connection);
            if(commandParameters !=null)
            {
                foreach(SqlParameter parm in commandParameters)
                    command.Parameters.Add(parm);
            }
            return command.ExecuteNonQuery();
        }
    }
//获取单个值
    public static object ExecuteScalar(string connectionString, CommandType commandType,
    string commandText, params SqlParameter[] commandParameters)
    {
        using(SqlConnection connection=new SqlConnection(connectionString))
        {
            if(connection.State !=ConnectionState.Open)
                connection.Open();
            SqlCommand command=new SqlCommand(commandText, connection);
            if(commandParameters !=null)
            {
                foreach(SqlParameter parm in commandParameters)
                    command.Parameters.Add(parm);
            }
            return command.ExecuteScalar();
        }
```

```
        }
//读取记录
        public static SqlDataReader ExecuteReader(string connectionString, CommandType
        commandType, string commandText, params SqlParameter[]
        commandParameters)
        {
            SqlConnection connection=new SqlConnection(connectionString);
            try
            {
                if(connection.State !=ConnectionState.Open)
                    connection.Open();
                SqlCommand command=new SqlCommand(commandText, connection);
                if(commandParameters !=null)
                {
                    foreach(SqlParameter parm in commandParameters)
                        command.Parameters.Add(parm);
                }
                SqlDataReader   rdr   =   command.   ExecuteReader   ( CommandBehavior.
                CloseConnection);
                return rdr;
            }
            catch
            {
                connection.Close();
                throw;
            }
        }
    }
```

3.2.3 多层 B/S 下实现数据访问

【示例 3.1】 搭建系统框架，利用三层架构实现下述功能：输入用户名，查询用户密码，实现数据在三层之间的传递。

【步骤 1】搭建系统架构，如图 3.4 所示。

【步骤 2】添加各层之间的依赖关系，如图 3.5～图 3.7 所示。

【步骤 3】添加数据库及数据表。

新建数据库 dbMMChaXun，添加数据表 UserCXs，如图 3.8 所示。



图 3.4 搭建系统架构



图 3.5　表示层添加引用

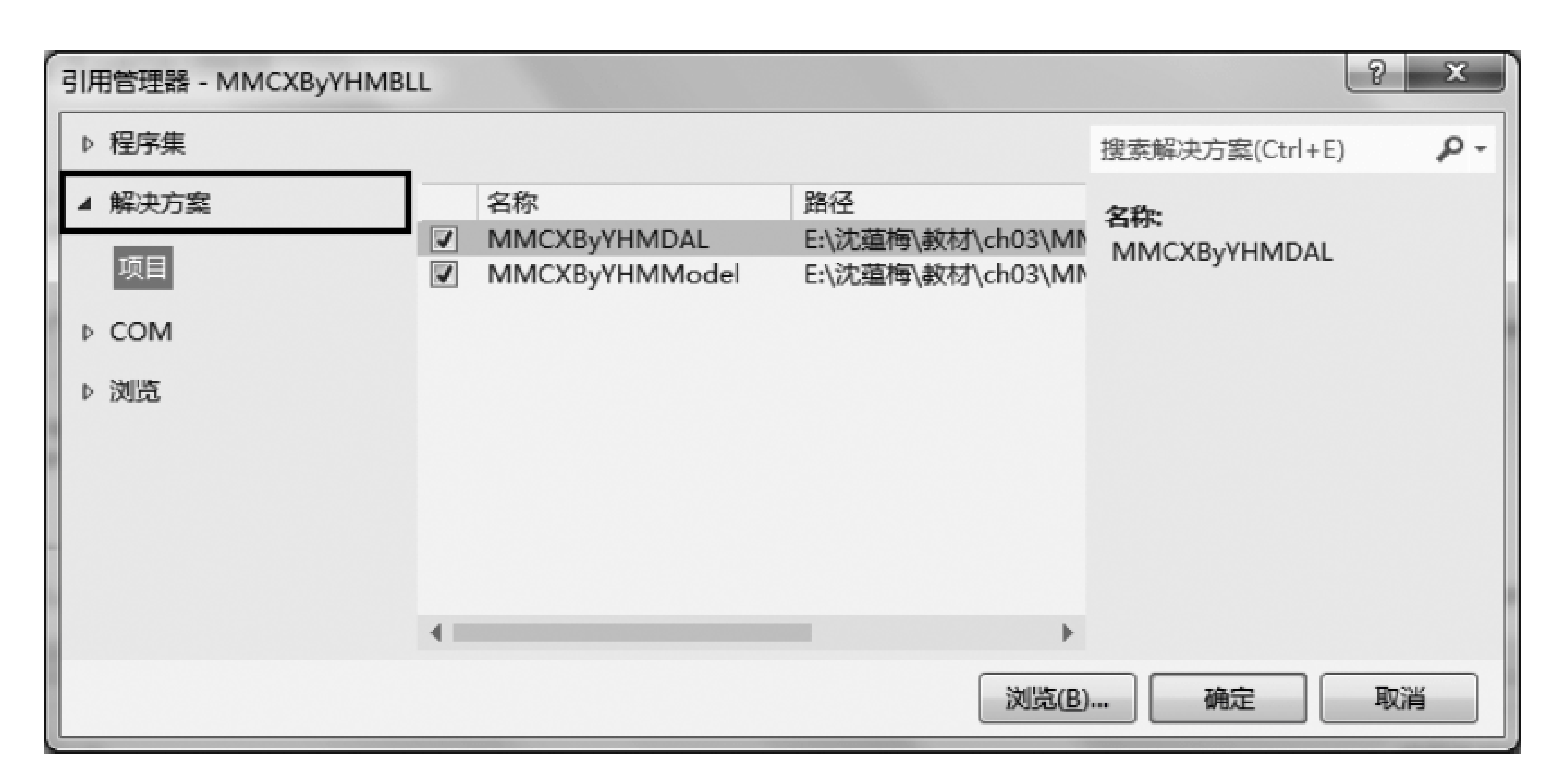


图 3.6　业务逻辑层添加引用



图 3.7　数据访问层添加引用

OEM-20130723TNF....un - dbo.UserCXs

列名	数据类型	允许 Null 值
UserName	nvarchar(50)	☑
UserPwd	nvarchar(50)	☑

图 3.8 数据表 UserCXs

【步骤 4】根据数据表 UserCXs 添加实体类 UserCX.cs，编写代码如下：

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace MMCXByYHMModel
{
    public class UserCX
    {
        private string userName;
        public string UserName
        {
            get { return userName; }
            set { userName=value; }
        }
        private string userPwd;
        public string UserPwd
        {
            get { return userPwd; }
            set { userPwd=value; }
        }
    }
}
```

【步骤 5】配置 Web.config。

```
<connectionStrings>
    <add name="aa" providerName="System.Data.SqlClient" connectionString=
    "Data Source=OEM-20130723TNF;Initial Catalog=dbMMChaXun;User ID=sa;Password=
   123456" />
  </connectionStrings>
```

【步骤 6】添加数据访问类 DBHelper.cs，并编写代码。见上节内容。

【步骤 7】数据访问层添加类 UserCXService.cs，并编写代码。

```
using System;
using System.Collections.Generic;
```

```
using System.Data;
using System.Data.SqlClient;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
namespace MMCXByYHMDAL
{
    public class UserCXService
    {
        public string GetMMByYHM(string YHM)
        {
            string sql="select UserPwd from dbo.UserCXs where UserName=
            @ UserName";
            return DBHelper.ExecuteScalar(DBHelper.ConnectionString, CommandType.Text, sql,
            new SqlParameter("@UserName", YHM)).ToString();
        }
    }
}
```

【步骤 8】业务逻辑层添加类 UserCXManager.cs,并编写代码。

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using MMCXByYHMDAL;
namespace MMCXByYHMBLL
{
    public class UserCXManager
    {
        public string GetMMByYHM(string YHM)
        {
            return new UserCXService().GetMMByYHM(YHM);
        }
    }
}
```

【步骤 9】表示层添加新项 MMCX.aspx,在源视图下编写代码如下:

```
<table>
        <tr><td>请输入用户名</td><td>
            <asp:TextBox ID="txtUserName" runat="server"></asp:TextBox></td> </tr>
        <tr><td colspan="2">
            <asp:Button ID="btnCX" runat="server" Text="查询" OnClick="btnCX_Click" /
            ></td></tr>
```

```
            <tr><td>查询结果密码为:</td><td>
                <asp:TextBox ID="txtPwd" runat="server"></asp:TextBox></td></tr>
        </table>
```

【步骤 10】在后台编写代码如下,页面浏览如图 3.9 所示。

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.UI;
using System.Web.UI.WebControls;
using MMCXByYHMBLL;
public partial class MMCX : System.Web.UI.Page
{
    protected void Page_Load(object sender, EventArgs e)
    {    }
    protected void btnCX_Click(object sender, EventArgs e)
    {
        txtPwd.Text=new UserCXManager().GetMMByYHM(txtUserName.Text.Trim());
    }
}
```

请输入用户名 aa
查询
查询结果密码为: 123

图 3.9 根据用户名查询密码页面

3.3 项 目 训 练

通过对以上内容的学习,了解了访问数据库的一般步骤,同时了解了数据在三层之间的传递方法,现在我们回到项目导入的任务中来。

【步骤 1】搭建系统架构,如图 3.10 所示。

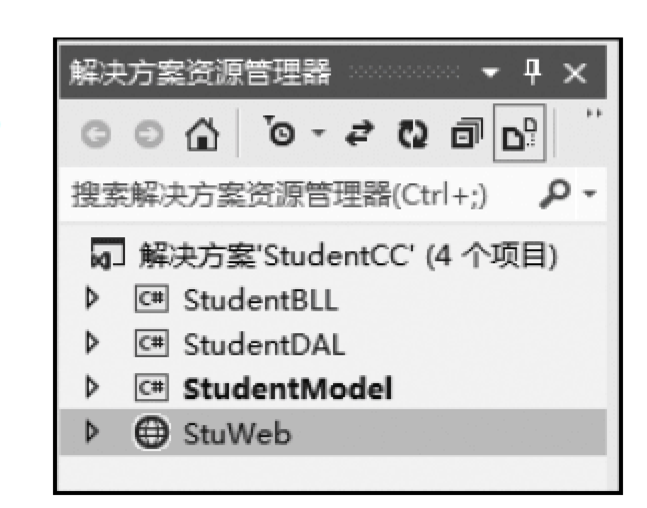


图 3.10 搭建系统架构

【步骤 2】添加各层之间的依赖关系,如图 3.11～图 3.13 所示。

【步骤 3】添加数据库及数据表。

新建数据库 dbStudent,添加数据表 tbStus,如图 3.14 所示。

【步骤 4】根据数据表 tbStus 添加实体类 tbStu.cs,编写代码如下:

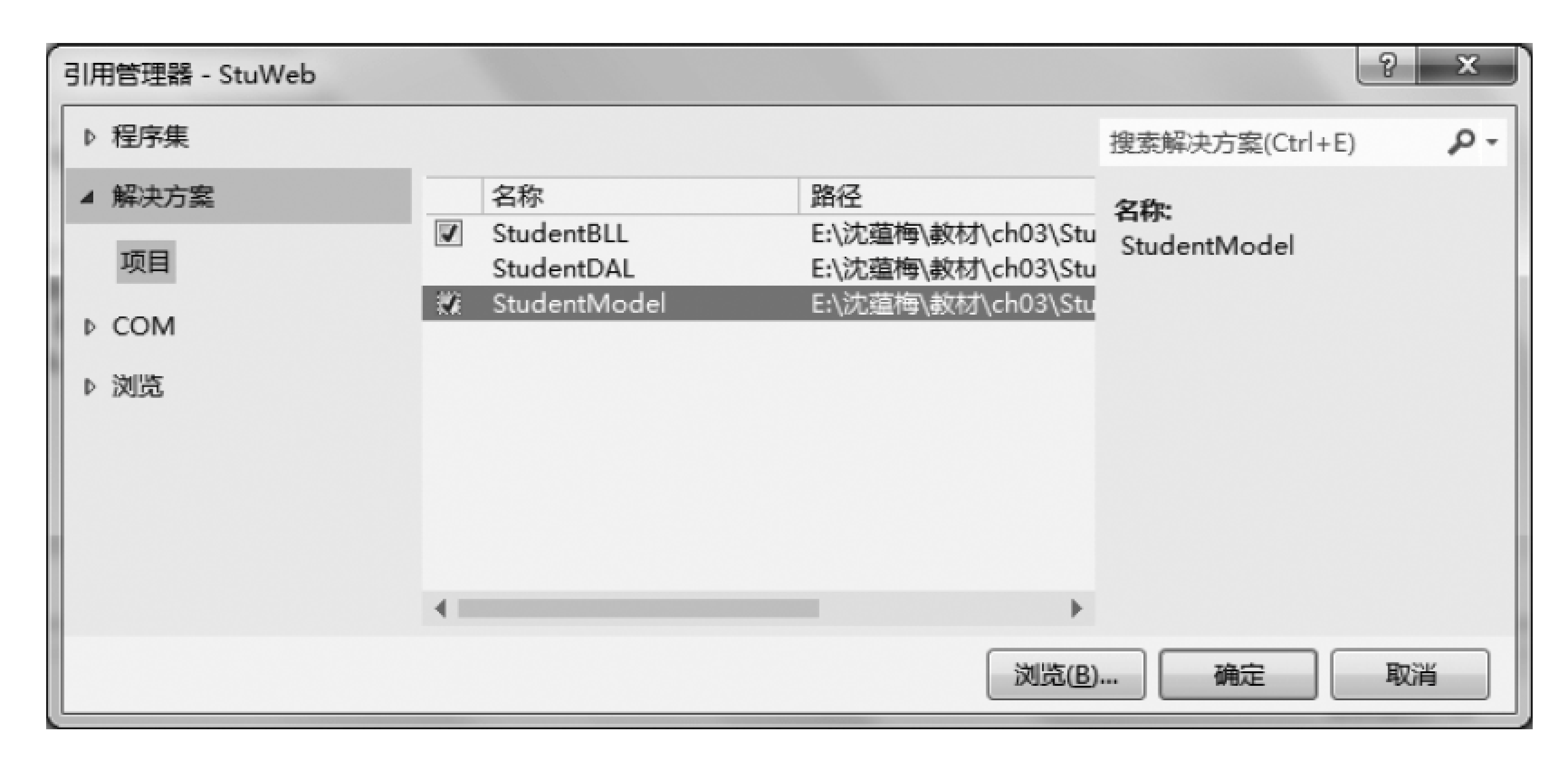


图 3.11　表示层添加引用



图 3.12　业务逻辑层添加引用



图 3.13　数据访问层添加引用

OEM-20130723TNF.d...dent - dbo.tbStus

列名	数据类型	允许 Null 值
Id	nvarchar(50)	☐
Name	nvarchar(50)	☑
Department	nvarchar(50)	☑
BanJi	nvarchar(50)	☑
Sex	nchar(10)	☑
SuShe	nchar(10)	☑
Tel	nvarchar(50)	☑
JiGuan	nvarchar(50)	☑
BYXX	nvarchar(50)	☑

图 3.14 数据表 tbStus

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace StudentModel
{
    public class tbStu
    {
        private string id;
        public string Id
        {
            get { return id; }
            set { id=value; }
        }
        private string name;
        public string Name
        {
            get { return name; }
            set { name=value; }
        }
        private string department;
        public string Department
        {
            get { return department; }
            set { department=value; }
        }
        private string banJi;
        public string BanJi
        {
            get { return banJi; }
            set { banJi=value; }
        }
```

```
        private string sex;
        public string Sex
        {
            get { return sex; }
            set { sex=value; }
        }
        private string suShe;
        public string SuShe
        {
            get { return suShe; }
            set { suShe=value; }
        }
        private string tel;
        public string Tel
        {
            get { return tel; }
            set { tel=value; }
        }
        private string jiGuan;
        public string JiGuan
        {
            get { return jiGuan; }
            set { jiGuan=value; }
        }
        private string bYXX;
        public string BYXX
        {
            get { return bYXX; }
            set { bYXX=value; }
        }
    }
}
```

【步骤 5】配置 Web. config。

```
<connectionStrings>
    <add name="aa" providerName="System.Data.SqlClient" connectionString= "Data Source
    =OEM-20130723TNF;Initial Catalog=dbStudent;User ID=sa;Password=123456" />
  </connectionStrings>
```

【步骤 6】添加数据访问类 DBHelper. cs，并编写代码。见上节内容。

【步骤 7】数据访问层添加类 tbStuService. cs，并编写代码。

```
using System;
using System.Collections.Generic;
using System.Linq;
```

```
using System.Text;
using System.Threading.Tasks;
using System.Data.SqlClient;
using System.Data;
using StudentModel;

namespace StudentDAL
{
    public class tbStuService
    {
        public tbStu  GetStudentById(string id)
        {
            string sql="select * from dbo.tbStus where Id=@Id";
            SqlDataReader dr=DBHelper.ExecuteReader(DBHelper.ConnectionString,
            CommandType.Text,sql,new SqlParameter("@Id",id));
            tbStu stu=new tbStu();
            if(dr.Read())
            {
                stu.Id=id;
                stu.Name=Convert.ToString(dr["Name"]);
                stu.Department=Convert.ToString(dr["Department"]);
                stu.BanJi=Convert.ToString(dr["BanJi"]);
                stu.Sex=Convert.ToString(dr["Sex"]);
                stu.SuShe=Convert.ToString(dr["SuShe"]);
                stu.Tel=Convert.ToString(dr["Tel"]);
                stu.JiGuan=Convert.ToString(dr["JiGuan"]);
                stu.BYXX=Convert.ToString(dr["BYXX"]);
            }
            return stu;
        }
    }
}
```

【步骤 8】业务逻辑层添加类 tbStuManager.cs,并编写代码。

```
using StudentModel;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using StudentDAL;

namespace StudentBLL
{
```

```
    public class tbStuManager
    {
        public tbStu GetStudentById(string id)
        {
            return new tbStuService().GetStudentById(id);
        }
    }
}
```

【步骤 9】表示层添加新项 StudentChaXun.aspx，在源视图下编写代码如下：

```
<table>
        <tr><td colspan="2" style="text-align:center">学生查询系统</td></tr>
        <tr><td>请输入学生学号:</td><td>
          <asp:TextBox ID="txtId" runat="server"></asp:TextBox><asp:Button ID="
          Button1" runat="server" Text="查询" OnClick="Button1_Click" />
          </td></tr>
        <tr><td>姓名</td><td>
            <asp:TextBox ID="txtName" runat="server" ReadOnly="True"></asp:TextBox>
           </td></tr>
        <tr><td>系部</td><td>
            <asp:TextBox ID="txtXB" runat="server" ReadOnly="True"></asp:TextBox></
            td></tr>
        <tr><td>班级</td><td>
            <asp:TextBox ID="txtBanJi" runat="server" ReadOnly="True"></asp:TextBox>
           </td></tr>
        <tr><td>宿舍</td><td>
            <asp:TextBox ID="txtSuShe" runat="server" ReadOnly="True"></asp:TextBox>
           </td></tr>
        <tr><td>性别</td><td>
            <asp:TextBox ID="txtSex" runat="server" ReadOnly="True"></asp:TextBox></
            td></tr>
        <tr><td>电话</td><td>
            <asp:TextBox ID="txtTel" runat="server" ReadOnly="True"></asp:TextBox></
            td></tr>
        <tr><td>籍贯</td><td>
            <asp:TextBox ID="txtJG" runat="server" ReadOnly="True"></asp:TextBox></
            td></tr>
        <tr><td>毕业学校</td><td>
            <asp:TextBox ID="txtBYXX" runat="server" ReadOnly="True"></asp:TextBox>
           </td></tr>
    </table>
```

【步骤 10】在后台编写代码如下：

```
using System;
```

```
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.UI;
using System.Web.UI.WebControls;
using StudentModel;
using StudentBLL;

public partial class StudentChaXun : System.Web.UI.Page
{
    protected void Page_Load(object sender, EventArgs e)
    {

    }
    protected void Button1_Click(object sender, EventArgs e)
    {
        tbStu st=new tbStu();
        st=new tbStuManager().GetStudentById(txtId.Text.Trim());
        txtBanJi.Text=st.BanJi.Trim();
        txtBYXX.Text=st.BYXX.Trim();
        txtJG.Text=st.JiGuan.Trim();
        txtName.Text=st.Name.Trim();
        txtSex.Text=st.Sex.Trim();
        txtSuShe.Text=st.SuShe.Trim();
        txtTel.Text=st.Tel.Trim();
        txtXB.Text=st.Department.Trim();
    }
}
```

3.4 平行项目训练

1. 训练内容

根据项目训练内容，新增学生插入页面，实现学生信息的增加功能。

2. 训练目的

(1) 进一步训练和巩固学生对三层之间数据传递方法的理解；

(2) 使学生对数据库访问有一个比较深刻的印象和掌握。

3. 训练过程

【步骤1】在数据访问层 tbStuService.cs 类中添加代码，实现数据插入功能，代码如下：

```
public bool AddStudent(tbStu st)
        {
```

```
        string sql="insert into dbo.tbStus(Id,Name,Department,BanJi,Sex,SuShe,Tel,
        JiGuan,BYXX) values (@Id,@Name,@Department,@BanJi,@Sex,@SuShe,@Tel,@
        JiGuan,@BYXX)";
        SqlParameter[] para=new SqlParameter[]
        {
            new SqlParameter("@Id",st.Id),
            new SqlParameter("@Name",st.Name),
            new SqlParameter("@Department",st.Department),
            new SqlParameter("@BanJi",st.BanJi),
            new SqlParameter("@Sex",st.Sex),
            new SqlParameter("@SuShe",st.SuShe),
            new SqlParameter("@Tel",st.Tel),
            new SqlParameter("@JiGuan",st.JiGuan),
            new SqlParameter("@BYXX",st.BYXX),
        };
        return DBHelper.ExecuteNonQuery(DBHelper.ConnectionString, CommandType.
        Text, sql, para)>0;
    }
```

【步骤 2】在业务逻辑层 tbStuManager.cs 类中添加代码，如下所示：

```
public bool AddStudent(tbStu st)
    {
        return new tbStuService().AddStudent(st);
    }
```

【步骤 3】表示层添加新项 AddStu.aspx，在源视图下编写代码如下：

```
<table>
    <tr><td colspan="2" style="text-align:center">学生增加系统</td></tr>
    <tr><td>学号</td><td>
       <asp:TextBox ID="txtId" runat="server"></asp:TextBox></td></tr>
    <tr><td>姓名</td><td>
        <asp:TextBox ID="txtName" runat="server"></asp:TextBox></td></tr>
    <tr><td>系部</td><td>
        <asp:TextBox ID="txtXB" runat="server"></asp:TextBox></td></tr>
    <tr><td>班级</td><td>
        <asp:TextBox ID="txtBanJi" runat="server"></asp:TextBox></td></tr>
    <tr><td>宿舍</td><td>
        <asp:TextBox ID="txtSuShe" runat="server"></asp:TextBox></td></tr>
    <tr><td>性别</td><td>
        <asp:TextBox ID="txtSex" runat="server"></asp:TextBox></td></tr>
    <tr><td>电话</td><td>
        <asp:TextBox ID="txtTel" runat="server"></asp:TextBox></td></tr>
    <tr><td>籍贯</td><td>
        <asp:TextBox ID="txtJG" runat="server"></asp:TextBox></td></tr>
```

```
        <tr><td>毕业学校</td><td>
            <asp:TextBox ID="txtBYXX" runat="server"></asp:TextBox></td></tr>
        <tr><td colspan="2" style="text-align:center"><asp:Button ID= "Button1" runat
        ="server" Text="插入" OnClick="Button1_Click" /></td></tr>
    </table>
```

【步骤4】在后台编写代码如下：

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.UI;
using System.Web.UI.WebControls;
using StudentBLL;
using StudentModel;

public partial class AddStu : System.Web.UI.Page
{
    protected void Page_Load(object sender, EventArgs e)
    {
    }
    protected void Button1_Click(object sender, EventArgs e)
    {
        tbStu st=new tbStu();
        st.Id=txtId.Text;
        st.BanJi=txtBanJi.Text;
        st.BYXX=txtBYXX.Text;
        st.JiGuan=txtJG.Text;
        st.Name=txtName.Text;
        st.Sex=txtSex.Text;
        st.SuShe =txtSuShe.Text;
        st.Tel=txtTel.Text;
        st.Department=txtXB.Text;
        if(new tbStuManager().AddStudent(st))
            Response.Write("插入成功");
        else
            Response.Write("插入失败");
    }
}
```

【步骤5】程序运行如图3.15所示。

3.5 总　　结

本章介绍了如何建立系统、SQL Server2008数据库的连接与如何对数据进行增加、



图 3.15　程序运行结果

删除、修改和查询等操作，主要学习了 ADO. NET 中的对象与方法。通过项目训练“学生查询系统”和平行项目训练“学生增加系统”，整体训练了如何实现对数据库的访问以及如何进行数据的增删改查。

3.6　习　　题

1. ADO. NET 包含哪几种对象？
2. 如何创建数据库访问类？
3. 调用数据库访问类，实现登录功能，参考界面如图 3.16 所示。

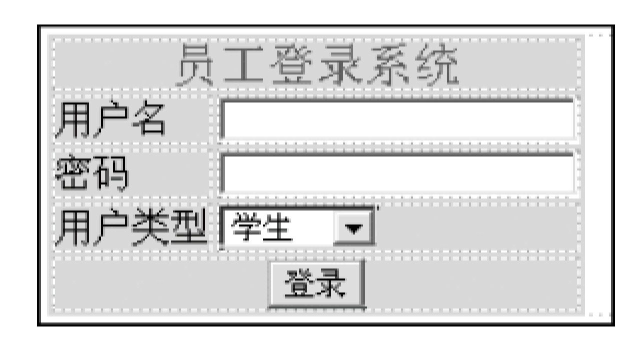


图 3.16　习题 3 参考界面

第 4 章　动态 Web 界面设计与编码

本章要点：

- 界面布局与设计
- 基于控件的详细设计
- 界面之间的调用与实现

技能目标：

- 会设计友好的用户界面
- 会设计用户界面中的控件
- 会在界面之间实现互相调用

4.1　项目导入

【项目场景】

苏州健雄职业技术学院要开发一个软件测评与外包科技服务平台，请你为该单位开发该平台，注册界面如图 4.1 所示。

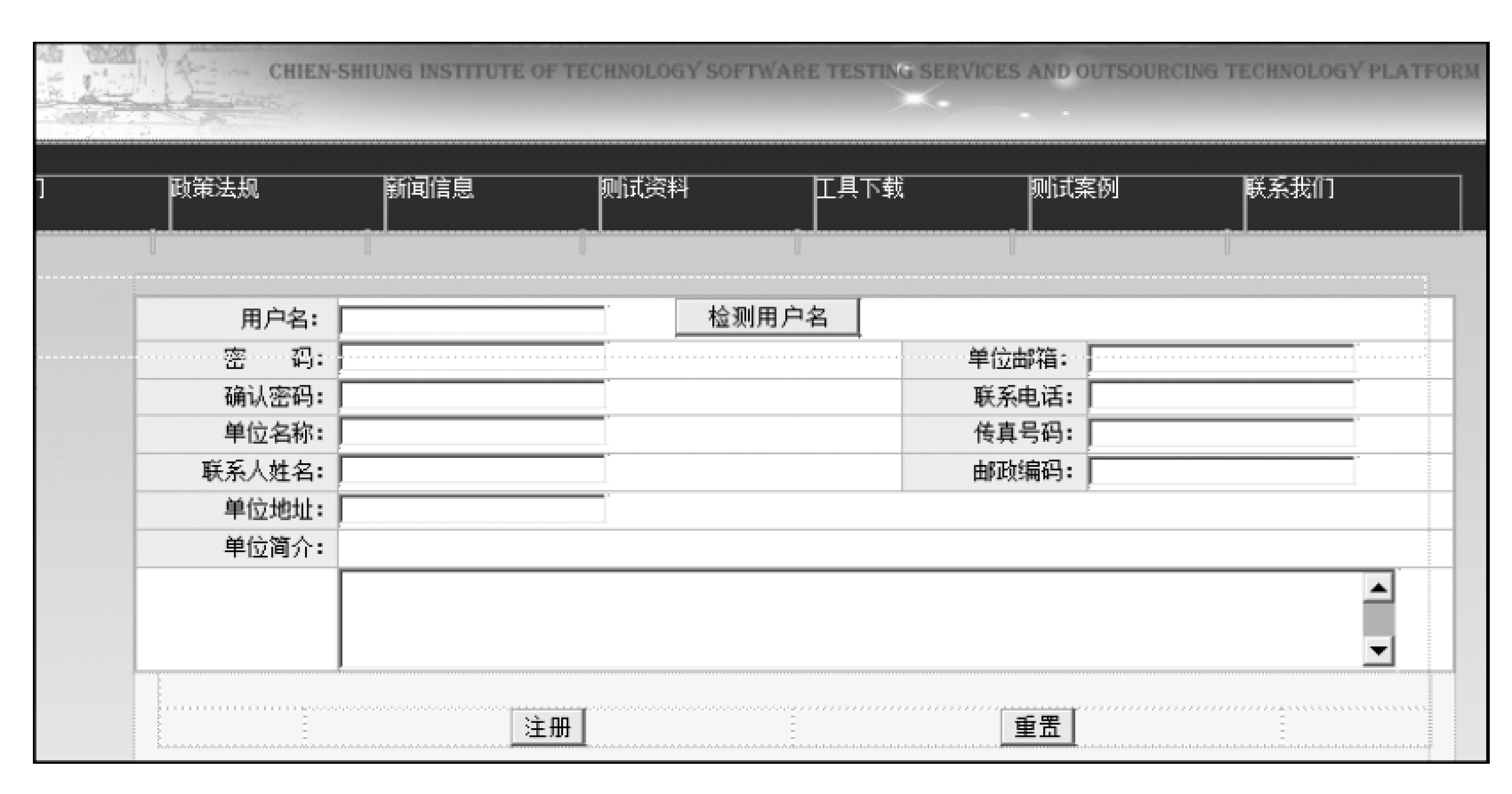


图 4.1　用户注册界面

【问题引导】

（1）如何设计用户注册界面。
（2）如何设置界面中的控件。

4.2　技术与知识准备

4.2.1　界面布局与设计

在进行界面布局时，应先针对在整个网站的网页中重复出现的部分，如网页顶部的图片、导航以及网页底部的文字等，分析是否有必要将其做在母版页中。母版页是一个以“.master”作为后缀名的文件，它首先将页面上的公用元素（如网站 Logo、广告条、导航条等）整合在一起。何为母版页，顾名思义母版就是模版，就像在 PPT 里面的板式或主题一样，大框架已经有了，我们的任务就是向里面添加具体的内容。这样我们制作的所有幻灯片的外观大体都是一样的。在 ASP.NET 中母版页有两种作用，一是提高代码的复用（把相同的代码抽出来），二是使整个网站保持一致的风格和样式。

母版页的使用与普通页面类似，可以在其中放置文件或者图形、任何的 HTML 控件和 Web 控件，后置代码等。与普通页面不一样的是，它可以包含 ContentPlaceHolder 控件，ContentPlaceHolder 控件就是可以显示内容页面的区域。

母版页仅仅是一个页面模板，单独的母版页是不能被用户所访问的。单独的内容页也不能够使用。母版页和内容页有严格对应关系。母版页包含多少个 ContentPlaceHolder 控件，那么内容页中也必须设置与其相对应的 Content 控件。当客户端浏览器向服务器发出请求，要求浏览某个内容页面时，引擎将同时执行内容页和母版页的代码，并将最终结果发送给客户端浏览器。

【示例 4.1】　新建母版页 houtaiMaster.master。

【步骤 1】搭建系统架构，并添加各层之间的依赖关系，如图 4.2 所示。



图 4.2　搭建系统框架

【步骤 2】右击 Dorm_OA，依次选择“添加”→“新建文件夹”，取名为“houtai”，右击“houtai”文件夹，依次选择“添加”→“添加新项”，弹出如图 4.3 所示的对话框。

【步骤 3】左侧选择 C＃，右侧选择母版页，下侧输入母版页名称 houtaiMaster.master，单击【添加】按钮，如图 4.4 所示。



图 4.3 【添加新项】对话框



图 4.4 添加母版页

【步骤 4】新创建的母版页上面默认有两个 ContentPlaceHolder，分别是预留给内容页的头部和主体部分显示的控件，可以把每个 ContentPlaceHolder 控件理解成一个"内容占位符"，至于每个占位符中到底会显示哪些内容，则由具体的内容页要显示的内容决定。创建后的母版页代码如下所示：

```
<%@ Master Language="C#" AutoEventWireup="true" CodeFile="houtaiMaster.master.cs"
Inherits="houtai_houtaiMaster" %>
<!DOCTYPE html>
<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
```

```
<meta http-equiv="Content-Type" content="text/html; charset=utf-8"/>
    <title></title>
    <asp:ContentPlaceHolder id="head" runat="server">
    </asp:ContentPlaceHolder>
</head>
<body>
    <form id="form1" runat="server">
    <div>
        <asp:ContentPlaceHolder id="ContentPlaceHolder1" runat="server">
        </asp:ContentPlaceHolder>
    </div>
    </form>
</body>
</html>
```

4.2.2　基于控件的详细设计

在母版页新建完成后，就可以根据需求制作具体母版页了，摆放相关控件，并设置控件的属性。

【示例 4.2】　完成母版页制作，如图 4.5 所示。



图 4.5　母版页

【步骤 1】在母版页中添加代码，如下所示：

```
<%@ Master Language="C#" AutoEventWireup="true" CodeFile="houtaiMaster.master.cs"
Inherits="houtai_houtaiMaster" %>
<!DOCTYPE html>
<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
<meta http-equiv="Content-Type" content="text/html; charset=utf-8"/>
    <title></title>
    <asp:ContentPlaceHolder id="head" runat="server">
```

```
        </asp:ContentPlaceHolder>
    </head>
    <body>
        <form id="form1" runat="server">
        <div>
            <table style="width:1000px;height:500px">
                <tr><td colspan="2">
                    <asp:Image ID="Image1" runat="server" ImageUrl="~/Images/index1.jpg"
                    /></td></tr>
                <tr><td style="height:300px; width:300px;vertical-align:top">
                    <asp:TreeView ID="TreeView1" runat="server">
                    </asp:TreeView>
                </td><td style="vertical-align:top; width:700px"><asp:ContentPlaceHolder
                id="ContentPlaceHolder1" runat="server">
            </asp:ContentPlaceHolder></td></tr>
                <tr><td colspan="2" style="text-align:center">版权所有:苏州健雄职业技
                术学院     地址:江苏省太仓市科教新城健雄路1号<br />
                邮政编码:215411     联系电话:0512- 53941111</td></tr
                >
            </table>
         </div>
        </form>
    </body>
    </html>
```

【步骤 2】TreeView 控件是一种用于显示层次化结构的控件，可用于显示树形导航，TreeView 控件的使用方式分为两种：视图中编辑和动态添加，这里我们只介绍视图中编辑。

（1）在“设计”视图中，右击 TreeView 控件，选择“编辑节点”。

（2）在“TreeView 节点编辑器”中的“节点”下，可以选择“添加根节点”，也可以选择“子节点”，在“TreeView 节点编辑器”中的“属性”下可以对当前选中的节点进行属性设置，如图 4.6 所示。

TreeView 的节点 TreeNode 常用属性见表 4.1。

表 4.1　TreeView 的节点 TreeNode 常用属性

属　　性	说　　明
Text	节点中显示的文本
Value	可用于存储节点的非显示数值
NavigateUrl	单击节点时导航到的 URL

（3）在视图中添加节点，并设置相应属性，如图 4.7 所示。

相应的源视图的代码如下所示：



图 4.6　TreeView 节点编辑器

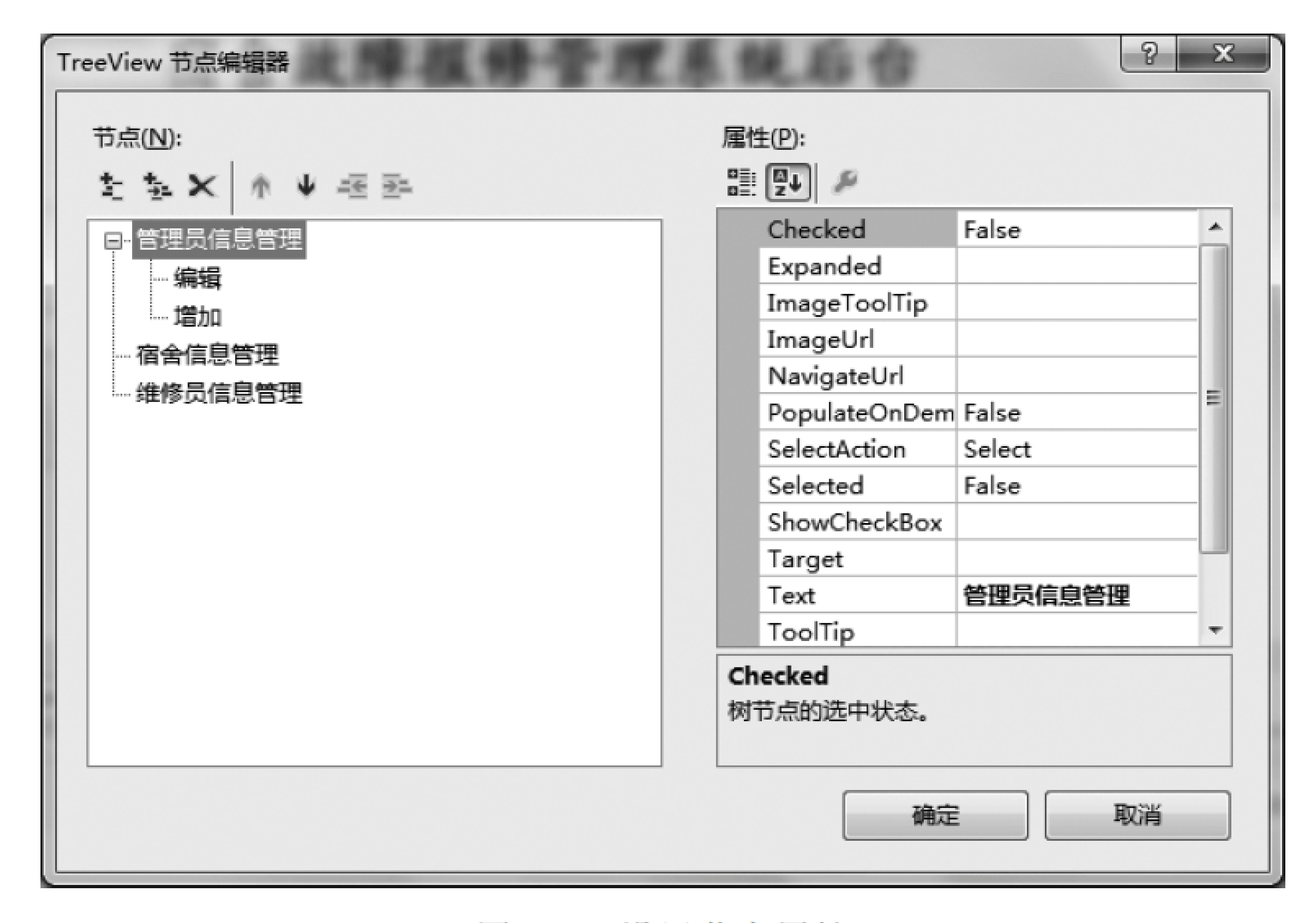


图 4.7　设置节点属性

```
<asp:TreeView ID="TreeView1" runat="server">
    <Nodes>
        <asp:TreeNode Text="管理员信息管理" Value="管理员信息管理">
            <asp:TreeNode Text="编辑" Value="编辑" NavigateUrl=
            "~/houtai/AdminEdit.aspx"></asp:TreeNode>
            <asp:TreeNode Text="增加" Value="增加" NavigateUrl=
            "~/houtai/AdminAdd.aspx"></asp:TreeNode>
        </asp:TreeNode>
        <asp:TreeNode Text="宿舍信息管理" Value="宿舍信息管理"></asp:TreeNode>
        <asp:TreeNode Text="维修员信息管理" Value="维修员信息管理"></asp:TreeNode>
    </Nodes>
</asp:TreeView>
```

4.2.3　界面之间的调用与实现

【示例 4.3】　基于母版页添加 AdminAdd.aspx 管理员增加页面，界面如图 4.8 所示。

图 4.8　管理员增加界面

【步骤 1】在解决方案资源管理器中，右击 houtai，选择“添加”→“添加新项”，弹出如图 4.9 所示的对话框，左侧选择 C＃，右侧选择 Web 窗体，在下面输入名称“AdminAdd.aspx”，选择母版页前面把钩勾上，单击“添加”按钮。



图 4.9　添加新项

【步骤 2】弹出选择母版页对话框，左侧项目文件夹选择 houtai，右侧文件夹内容选择“houtaiMaster.master”，单击“确定”按钮，如图 4.10 所示。

【步骤 3】基于母版页生成的内容页“AdminAdd.aspx”，源视图代码如下所示：

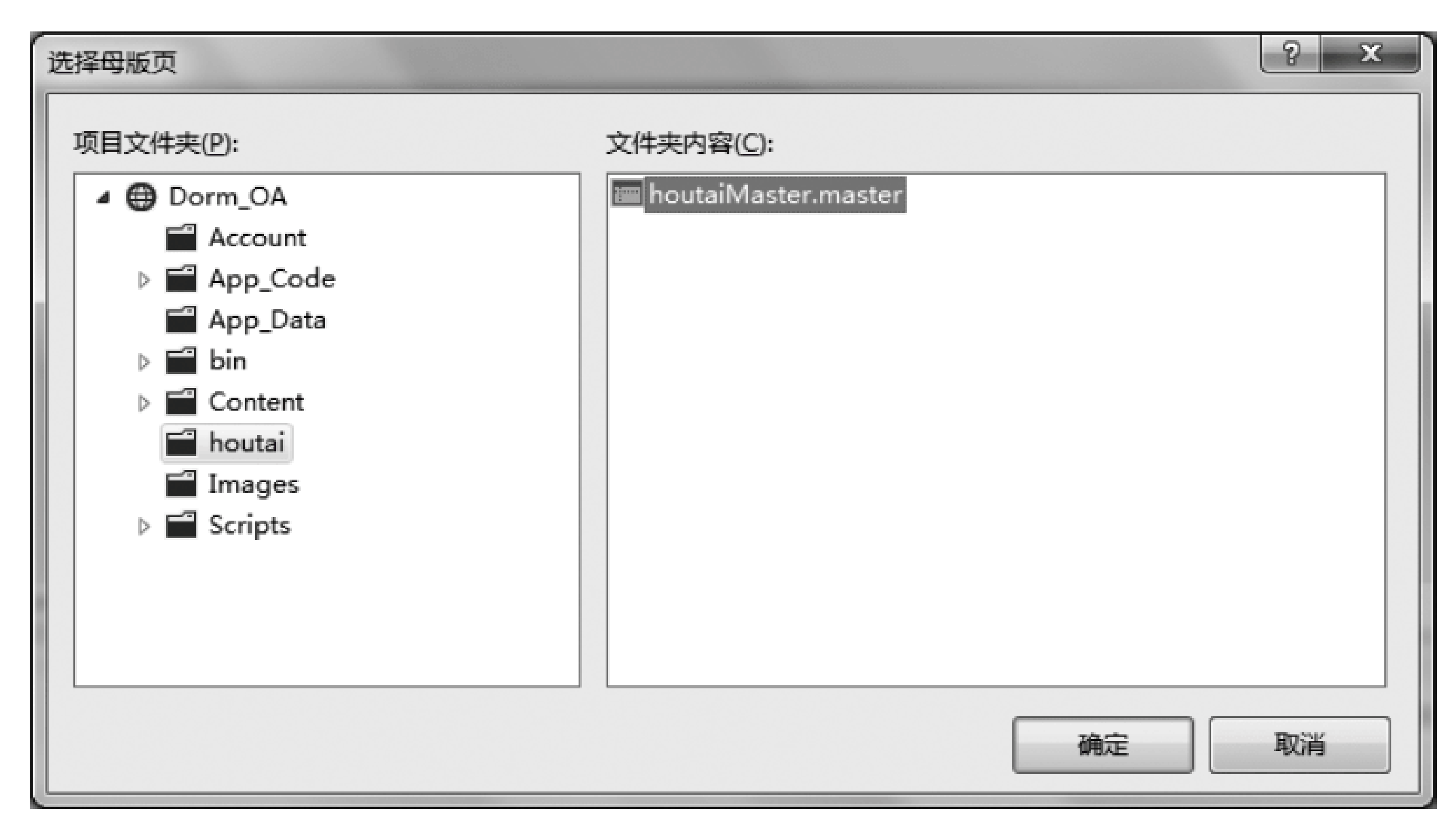


图 4.10　选择母版页

```
<%@ Page Title="" Language="C#" MasterPageFile="~/houtai/houtaiMaster.master"
AutoEventWireup="true" CodeFile="AdminAdd.aspx.cs" Inherits="houtai_AdminAdd" %>
<asp:Content ID="Content1" ContentPlaceHolderID="head" Runat="Server">
</asp:Content>
<asp:Content ID="Content2" ContentPlaceHolderID="ContentPlaceHolder1" Runat=
"Server">
</asp:Content>
```

【步骤 4】在内容页中添加两个文本框(TextBox)和一个按钮(Button),并修改相应属性,源视图代码如下所示:

```
<%@ Page Title="" Language="C#" MasterPageFile="~/houtai/houtaiMaster.master"
AutoEventWireup="true" CodeFile="AdminAdd.aspx.cs" Inherits="houtai_AdminAdd" %>
<asp:Content ID="Content1" ContentPlaceHolderID="head" Runat="Server">
</asp:Content>
<asp:Content ID="Content2" ContentPlaceHolderID="ContentPlaceHolder1" Runat=
"Server">
    <table><tr style="height:40px"><td>用户名</td><td>
        <asp:TextBox ID="TextBox1" runat="server"></asp:TextBox></td></tr>
        <tr style="height:40px"><td>密码</td><td>
        <asp:TextBox ID="TextBox2" runat="server"></asp:TextBox></td></tr>
        <tr style="height:40px"><td colspan="2" style="text-align:center">
            <asp:Button ID="Button1" runat="server" Text="添加" /></td></tr>
    </table>
</asp:Content>
```

4.3　项 目 训 练

通过对以上内容的学习,了解了创建母版页的步骤,同时了解了控件属性的设置方法

以及界面调用的方法，现在我们回到项目导入的任务中来。

【步骤 1】搭建系统框架，添加各层之间的依赖关系，如图 4.11 所示。

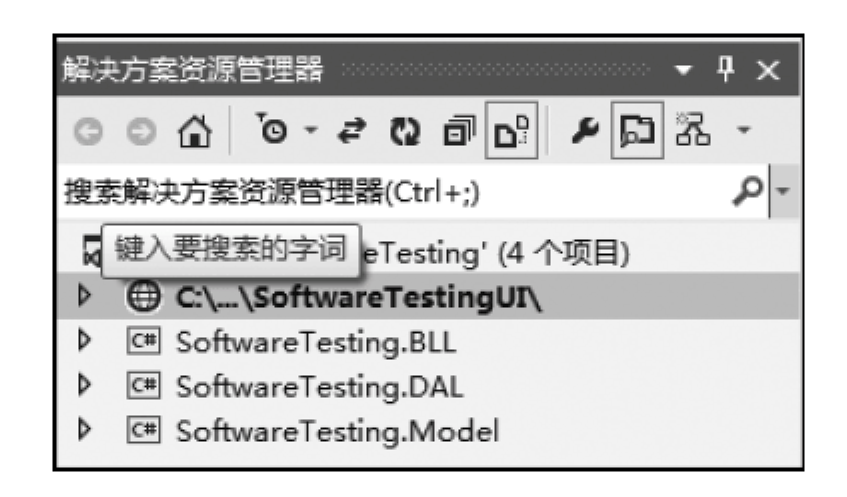


图 4.11　搭建系统框架

【步骤 2】新建一个母版页，右击 Web，依次选择"添加"→"添加新项"，在弹出的窗口中，选择"母版页"，然后修改名称为 top.master，单击"添加"按钮，如图 4.12 所示。



图 4.12　新建母版页 top.master

【步骤 3】在母版页中添加代码，完成母版页制作，如图 4.13 所示。

```
<%@ Master Language="C#" AutoEventWireup="true" CodeFile="Top.master.cs" Inherits="Top"
%>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/
xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
    <title></title>
    <asp:ContentPlaceHolder id="head" runat="server">
    </asp:ContentPlaceHolder>
    <style type="text/css">
```



图 4.13　母版页 top.master

```
.menu{background:url('images/bg_2.gif')repeat-x;
height:52px;padding-left:80px;   }
 * {margin:0px; padding:0px;}
.menusel{ float:left;width:115px; position:relative; margin-top:20px;}
.menusel h2{ font-size:12px; height:32px; font-weight:normal;}
.menusel a{text-align:center; width:130px;height:25px;  position:relative; z-index:2;
color:#fff;}
a{ color:#687f96; text-decoration:none;}
.position{ position:absolute; z-index:1; margin-left:-10px;}
.menusel ul{width:130px; background:#608ab3;margin-top: -1px; position:relative; z-
index:1; display:none;}
.typeul{  margin-left:0;   }
.clearfix{display:inline-block;}
ul{list-style-type: none;}
.typeul li{border-bottom:1px solid #fff;width:124px; position:relative; float:left;
height:35px; line-height:35px; padding-left:6px;}
.menusel .lli{ border:none; }
    </style>
</head>
<body>
    <form id="form1" runat="server">
    <div>
        <img src="Images/bannernew1.jpg" /><div class="menu">
    <div class="menusel" style="width:95px;">
      <h2><a href="index.html">首页</a></h2>
    </div>
    <div id="menu1" class="menusel">
      <h2><a href="xiangqing.html">了解我们</a></h2>
      <div class="position">
        <ul class="clearfix typeul">
          <li><a href="http://www.865171.cn">公司简介</a></li>
          <li><a href="http://www.865171.cn">企业文化</a></li>
```

```
            <li class="lli"><a href="http://www.865171.cn">联系我们</a></li>
        </ul>
    </div>
</div>
<div id="menu2" class="menusel">
    <h2><a href="xiangqing.html">政策法规</a></h2>
    <div class="position">
        <ul class="clearfix typeul"></ul>
    </div>
</div>
<div id="menu3" class="menusel">
    <h2><a href="xinwen.html">新闻信息</a></h2>
    <div class="position">
        <ul class="clearfix typeul">
            <li><a href="http://www.865171.cn">行业新闻</a></li>
            <li><a href="http://www.865171.cn">公司新闻</a></li>
            <li class="lli"><a href="http://www.865171.cn">技术文摘</a></li>
        </ul>
    </div>
</div>
<div id="menu4" class="menusel">
    <h2><a href="http://www.865171.cn">测试资料</a></h2>
    <div class="position">
        <ul class="clearfix typeul">
            <li class="lli"><a href="http://www.865171.cn">我们的软件</a></li>
        </ul>
    </div>
</div>
<div id="menu5" class="menusel">
    <h2><a href="http://www.865171.cn">工具下载</a></h2>
    <div class="position">
        <ul class="clearfix typeul">
        </ul>
    </div>
</div>
<div id="menu6" class="menusel">
    <h2><a href="http://www.865171.cn">案例分析</a></h2>
    <div class="position">
        <ul class="clearfix typeul">
            <li><a href="http://www.865171.cn">发表留言</a></li>
            <li><a href="http://www.865171.cn">留言列表</a></li>
            <li><a href="http://www.865171.cn">留言</a></li>
            <li><a href="http://www.865171.cn">建议</a></li>
            <li class="lli"><a href="http://www.865171.cn">投诉</a></li>
```

```
        </ul>
      </div>
    </div>
    <div id="menu7" class="menusel">
      <h2><a href="http://www.865171.cn">联系我们</a></h2>
      <div class="position">
        <ul class="clearfix typeul">    </ul>
      </div>
    </div>
  </div>
        <asp:ContentPlaceHolder id="ContentPlaceHolder1" runat="server">
                </asp:ContentPlaceHolder>
    </div>
    </form>
</body>
</html>
```

【步骤4】右击 Web,依次选择“添加”→“添加新项”,如图4.14所示。

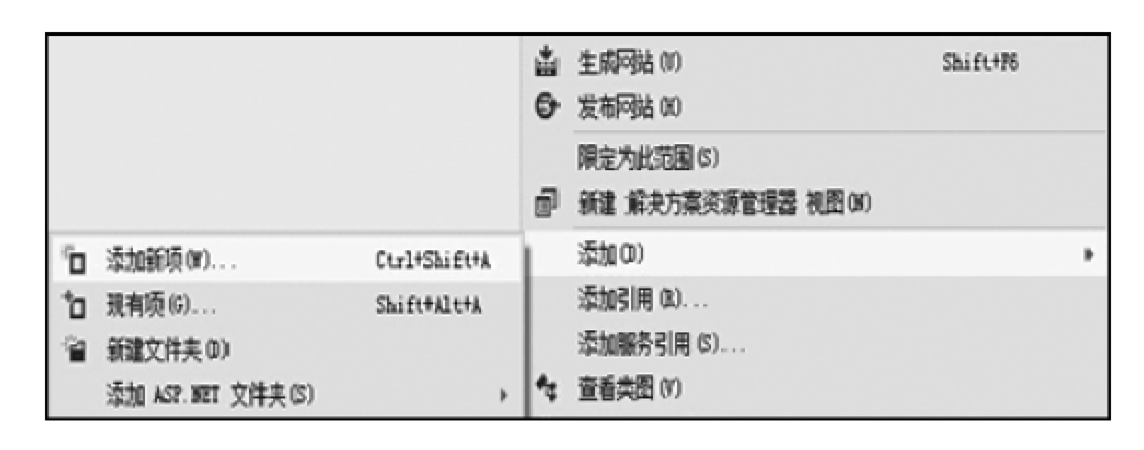


图4.14　选择“添加新项”

【步骤5】在弹出的对话框中选择“Web 窗体”,更改名称为:register.aspx,并勾选“选择母版页”选项,最后单击“添加”按钮,如图4.15所示。

【步骤6】在弹出的对话框中,选择之前制作好的 top.master 母版页,如图4.16所示。

【步骤7】在 register.aspx 页面中进行布局,拖动文本框与按钮,如图4.17所示。

【步骤8】创建数据库 WebSites,添加数据表 Company,如图4.18所示,并配置 Web.config:<connectionStrings>

```
  <add name="News" connectionString="server=YJATGJTKXXXUD5H; database=WebSites; uid=
 sa; pwd=123456" providerName="System.Data.SqlClient"/>
</connectionStrings>
```

【步骤9】根据数据表 Company,在实体层添加类 Company.cs,编写代码如下:

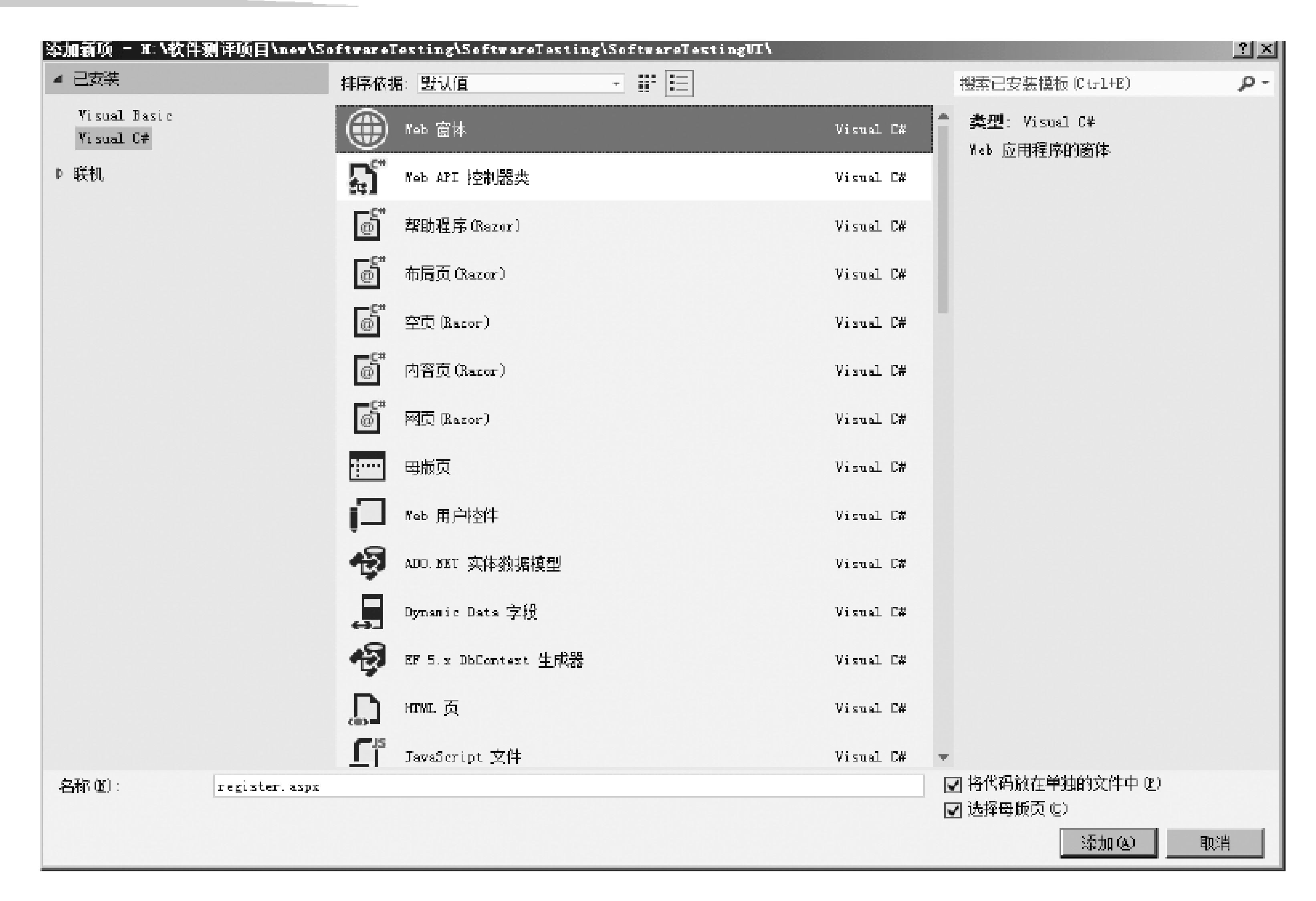


图 4.15 选择“Web 窗体”选项



图 4.16 选择“top. master”母版页



图 4.17 register. aspx 界面设计

OEM-20130723TNF....es - dbo.Company

列名	数据类型	允许 Null
Id	int	☐
UserName	nvarchar(MAX)	☑
PWD	nvarchar(MAX)	☑
CName	nvarchar(MAX)	☑
LXName	nvarchar(MAX)	☑
Address	nvarchar(MAX)	☑
CIntroduce	nvarchar(50)	☐
Email	nvarchar(MAX)	☐
Tel	nvarchar(MAX)	☑
Fax	nvarchar(MAX)	☐
Zip	nvarchar(MAX)	☐

图 4.18　Company 数据表

```
public class Company
    {
        private int id;
        public int Id
        {
            get { return id; }
            set { id=value; }
        }
        private string username;
        public string Username
        {
            get { return username; }
            set { username=value; }
        }
        private string pwd;
        public string Pwd
        {
            get { return pwd; }
            set { pwd=value; }
        }
        private string cName;
        public string CName
        {
            get { return cName; }
            set { cName=value; }
        }
        private string lxName;
        public string LxName
        {
            get { return lxName; }
            set { lxName=value; }
        }
```

```
            private string address;
            public string Address
            {
                get { return address; }
                set { address=value; }
            }
            private string cintroduce;
            public string Cintroduce
            {
                get { return cintroduce; }
                set { cintroduce=value; }
            }
            private string email;
            public string Email
            {
                get { return email; }
                set { email=value; }
            }
            private string tel;
            public string Tel
            {
                get { return tel; }
                set { tel=value; }
            }
            private string fax;
            public string Fax
            {
                get { return fax; }
                set { fax=value; }
            }
            private string zip;
            public string Zip
            {
                get { return zip; }
                set { zip=value; }
            }
        public Company(){}
    }
```

【步骤 10】添加数据访问类 SqlHelper.cs,并编写代码,见第 3 章内容。

【步骤 11】数据访问层添加类 CompanyService.cs,编写代码如下:

```
public void AddCompany(Company company)
    {
        string sql="INSERT Company(UserName, PWD, CName, LXName, Address, CIntroduce,
```

```
        Email,Tel,Fax,Zip)"+"VALUES(@UserName,@PWD,@CName,@LXName,@Address,@
        CIntroduce,@Email,@Tel,@Fax,@Zip)";
        sql+=";SELECT @@IDENTITY";
        SqlParameter[] para=new SqlParameter[]
        {
            new SqlParameter("@UserName",company.Username),
            new SqlParameter("@PWD",company.Pwd),
            new SqlParameter("@CName",company.CName),
            new SqlParameter("@LXName",company.LxName),
            new SqlParameter("@Address",company.Address),
            new SqlParameter("@CIntroduce",company.Cintroduce),
            new SqlParameter("@Email",company.Email),
            new SqlParameter("@Tel",company.Tel),
            new SqlParameter("@Fax",company.Fax),
            new SqlParameter("@Zip",company.Zip)
        };
        company.Id= Convert.ToInt32(SqlHelper.ExecuteScalar(this.connection,
        CommandType.Text, sql, para));
    }
```

【步骤 12】业务逻辑层编写代码如下：

```
public void AddCompany(Company company)
    {
        new CompanyService().AddCompany(company);
    }
```

【步骤 13】编写“注册”按钮代码：

```
protected void btnRegister_Click(object sender, EventArgs e)
    {
        Company company=new Company();
        company.Username=txtUserName.Text;
        company.Pwd=txtPassWord.Text;
        company.CName=txtCName.Text;
        company.LxName=txtLXName.Text;
        company.Address=txtAddress.Text;
        company.Cintroduce=txtCIntroduce.Text;
        company.Email=txtEmail.Text;
        company.Tel=txtTel.Text;
        company.Fax=txtFax.Text;
        company.Zip=txtZip.Text;
        CompanyManager manager=new CompanyManager();
        manager.AddCompany(company);
        Page.RegisterClientScriptBlock("alert", "<script>alert('注册成功!')
        </script>");
}
```

4.4 平行项目训练

1. 训练内容

界面布局与设计、基于控件的详细设计。

2. 训练目的

(1) 进一步训练和巩固学生对控件设计方法的理解;

(2) 使学生对友好的用户界面设计有一个比较深刻的印象和认识。

3. 训练过程

【步骤 1】创建 Web 窗体 DownLoadList. aspx,并使用母版页 Top. master,并在页面中添加 HyperLink 控件,如图 4.19 所示。



图 4.19 DownLoadList. aspx 界面设计

【步骤 2】设置 HyperLink 控件属性 ForeColors = Black(控件中文本颜色),NavigateUrl="~/DownLoadDetail. aspx"(定位到的 URL),Text=loadrunner8. 1(该链接显示的文本),如图 4.20 所示。

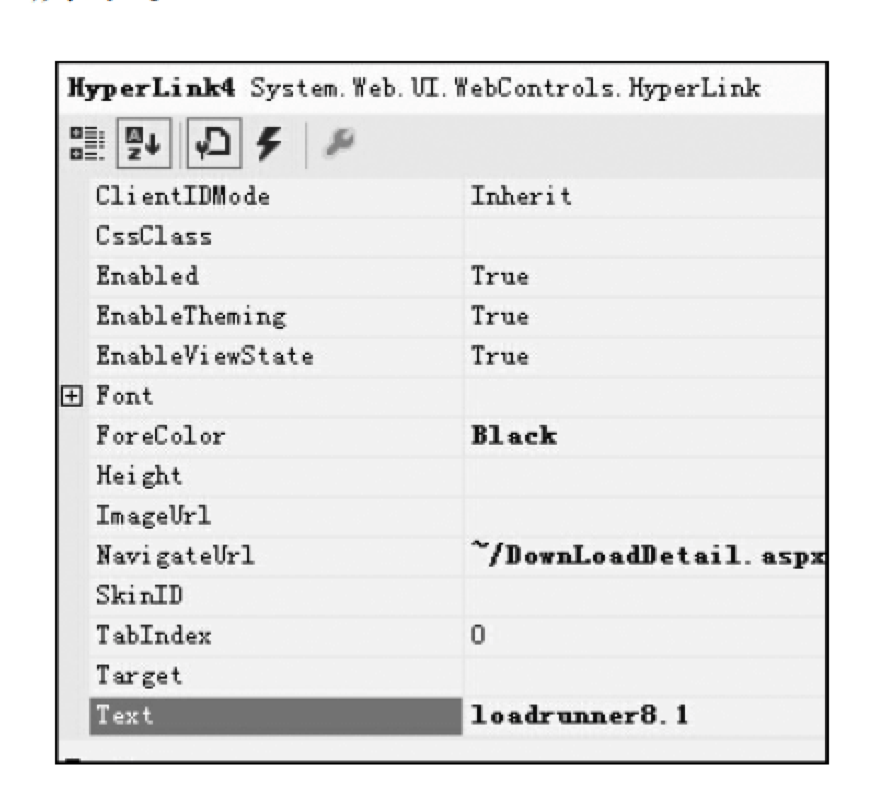


图 4.20 HyperLink 控件属性设计

4.5 总　　结

本章通过简单项目案例,介绍了如何进行界面布局与设计,如何对控件进行详细设计以及如何实现界面之间的调用。通过项目训练"注册界面"及平行项目"测试软件界面"整体训练了详细设计友好的用户界面的方法。

4.6 习　　题

1. 什么是母版页？母版页有什么作用?
2. TreeView 控件数据添加分为哪两种方式?
3. 创建母版页,如图 4.21 所示。



图 4.21　创建母版页

4. 基于母版页创建内容页,完成密码修改功能,界面如图 4.22 所示。

图 4.22　修改密码界面

第 5 章　数据绑定与高级编码

本章要点：

- 录入数据方法与编码
- 查询数据方法与编码
- 删除数据方法与编码
- 修改数据方法与编码

技能目标：

- 会编码获取 GridView 单元格的数据
- 会编码实现 GridView 基于单元格的更新
- 会编码实现 GridView 中的数据删除
- 会编码处理 GridView 的常用事件

5.1　项目导入

【项目场景】

太仓某公司要开发一个小型系统，能够实现该公司信息的增删改查功能，运行结果如图 5.1～图 5.3 所示，请你为该公司开发该系统，实现相应功能。

添加公司信息	
公司名称：	
地 址：	
邮政编码：	
联系电话：	
传 真：	
联 系 人：	
E-mail：	
	添加

图 5.1　添加公司信息

公司信息一览表				
公司名称	Email	联系人	电话	删除
新科技有限公司	mm@126.com	高女士	0512-6972266	删除
奇奇科技有限公司	ms@163.com	张女士	0512-6972266	删除
新世纪集团	mm@163.com	李小丽	0512-6972266	删除
家化集团	mq@163.com	杨柳	0512-6978981	删除
明天公司	yu@163.com	李小同	0512-6978981	删除

图 5.2　查询、删除公司信息

公司信息一览表					
公司名称	Email	联系人	电话		删除
新科技有限公司	my@126.com	gao女士	0512-6972266	更新 取消	删除
奇奇科技有限公司	ms@163.com	张女士	0512-6972266	编辑	删除
新世纪集团	mm@163.com	李小丽	0512-6972266	编辑	删除
家化集团	mq@163.com	杨柳	0512-6978981	编辑	删除

图 5.3　修改公司信息

【问题引导】

(1) 如何添加数据。
(2) 如何查询数据。
(3) 如何删除数据。
(4) 如何修改数据。

5.2　技术与知识准备

5.2.1　录入数据方法与编码

【示例 5.1】　新建一个学生管理系统，能够实现学生信息的添加功能，如图 5.4 所示。

【步骤 1】搭建系统框架，添加各层之间的依赖关系，如图 5.5 所示。

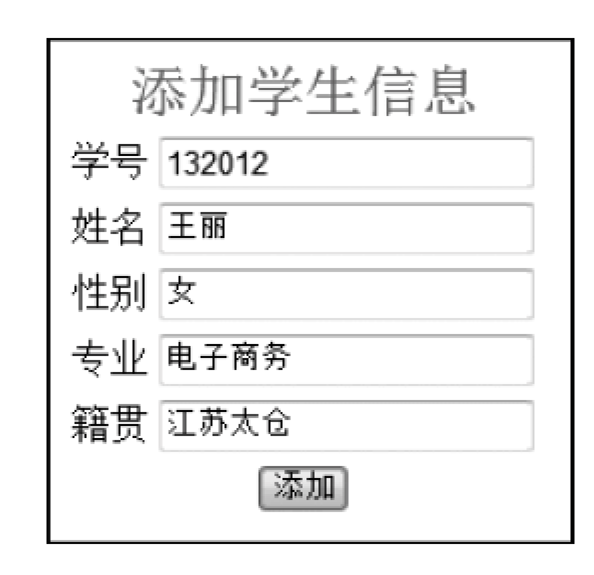


图 5.4　添加学生信息



图 5.5　搭建系统框架

【步骤 2】创建数据库 dbStu，添加数据表 tbStuInfos，如图 5.6 所示，并配置 Web.config：<connectionStrings>

```
<add name="DefaultConnection" providerName="System.Data.SqlClient" connectionString
="Data Source=.;Initial Catalog=dbStu;Integrated Security=True" />
</connectionStrings>
```

OEM-20130723TNF.d... - dbo.tbStuInfos

列名	数据类型	允许 Null 值
Sno	nvarchar(50)	☐
Sname	nvarchar(50)	☑
Sex	nchar(10)	☑
ZhuanYe	nvarchar(50)	☑
JiGuan	nvarchar(50)	☑

图 5.6　学生信息表

【步骤 3】根据数据表，在实体层添加类 tbStuInfo.cs，编写代码如下：

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
namespace StuModel
{
  public class tbStuInfo
    {
        private string sno;
        public string Sno
        {
            get { return sno; }
            set { sno=value; }
        }
        private string sname;
        public string Sname
        {
            get { return sname; }
            set { sname=value; }
        }
        private string sex;
        public string Sex
        {
            get { return sex; }
            set { sex=value; }
        }
        private string zhuanYe;
```

```
            public string ZhuanYe
            {
                get { return zhuanYe; }
                set { zhuanYe=value; }
            }
            private string jiGuan;
            public string JiGuan
            {
                get { return jiGuan; }
                set { jiGuan=value; }
            }
        }
}
```

【步骤 4】添加数据访问类 DBHelper.cs，并编写代码，见第 3 章内容。

【步骤 5】数据访问层添加类 tbStuInfoService.cs，编写代码如下：

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using StuModel;
using System.Data;

namespace StuDAL
{
    public class tbStuInfoService
      {
         public bool AddStuInfo(tbStuInfo tbstu)
         {
             string sql = string.Format("insert into tbStuInfos(Sno, Sname, Sex, ZhuanYe,
             JiGuan)values('{0}','{1}','{2}','{3}','{4}')", tbstu.Sno, tbstu.Sname, tbstu.
             Sex, tbstu.ZhuanYe, tbstu.JiGuan);
             return DBHelper.ExecuteNonQuery(DBHelper.ConnectionString, CommandType.Text,
             sql)>0;
         }
      }
}
```

【步骤 6】业务逻辑层编写代码如下：

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
```

```
using System.Threading.Tasks;
using StuModel;
using StuDAL;

namespace StuBLL
{
    public class tbStuInfoManager
    {
        public bool AddStuInfo(tbStuInfo tbstu)
        {
            return new tbStuInfoService().AddStuInfo(tbstu);
        }
    }
}
```

【步骤 7】右击 StuMWeb，添加页面 Reg.aspx，源视图代码如下：

```
<%@ Page Language="C#" AutoEventWireup="true" CodeFile="Reg.aspx.cs" Inherits="Reg" %>
<!DOCTYPE html>
<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
<meta http-equiv="Content-Type" content="text/html; charset=utf-8"/>
    <title></title>
</head>
<body>
    <form id="form1" runat="server">
    <div>
    <table>
        <tr><td colspan="2"  style="text-align:center; font-size:large;  color:red"
        >添加学生信息</td></tr>
         <tr><td>学号</td><td>
             <asp:TextBox ID="txtSno" runat="server"></asp:TextBox></td></tr>
         <tr><td>姓名</td><td><asp:TextBox ID="txtSname" runat="server">
         </asp:TextBox></td></tr>
         <tr><td>性别</td><td><asp:TextBox ID="txtSex" runat="server"></asp:
         TextBox></td></tr>
         <tr><td>专业</td><td><asp:TextBox ID="txtZhanYe" runat="server"></asp:
         TextBox></td></tr>
          <tr><td>籍贯</td><td><asp:TextBox ID="txtJiGuan" runat="server"></asp:
          TextBox></td></tr>
          <tr><td colspan="2" style="text-align:center">
              <asp:Button ID="Button1" runat="server" Text="添加" /></td></tr>
    </table>
    </div>
    </form>
```

```
</body>
</html>
```

【步骤 8】表示层后台代码如下：

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.UI;
using System.Web.UI.WebControls;
using StuBLL;
using StuModel;
public partial class Reg : System.Web.UI.Page
{
    protected void Page_Load(object sender, EventArgs e)
    {
    }
    protected void Button1_Click(object sender, EventArgs e)
    {
        tbStuInfo tbs=new tbStuInfo();
        tbs.Sno=txtSno.Text;
        tbs.Sname=txtSname.Text;
        tbs.Sex=txtSex.Text;
        tbs.ZhuanYe=txtZhanYe.Text;
        tbs.JiGuan=txtJiGuan.Text;
        if(new tbStuInfoManager().AddStuInfo(tbs))
            Response.Write("<script>alert('添加成功')</script>");
        else
            Response.Write("<script>alert('添加失败')</script>");
    }
}
```

5.2.2　查询数据方法与编码

ASP.NET 中有以下两种数据绑定方式：

1. 编码指定数据源

所谓的指定数据源的方式，就是编写代码在程序运行中动态绑定数据源，例如：

```
GridView1.DataSource=new UserManager().GetUsers();
//业务逻辑层已编写方法 GetUsers(),获取所有用户
GridView1.DataBind();
```

GridView1 就是数据绑定控件 GridView。

2. 使用数据源控件

数据源控件用于实现从不同数据源(数据库、XML 文件或业务逻辑层对象)获取数据的功能,它可以设置连接信息、查询信息、参数和行为,这样就可以把指定的数据绑定到数据绑定控件。常见的数据源控件有 SqlDataSource,XMLDataSource 等。

常见的数据绑定控件见表 5.1 所示。

表 5.1　常见的数据绑定控件

控件名称	用　途
DropDownList	下拉列表框,可以使用下拉菜单的形式供用户选择
GridView	通过表格方式实现数据的展示,其中每列表示一个字段,每行表示一条记录

【示例 5.2】 用编码指定数据源方式实现姓名模糊查询功能,见图 5.7 所示。

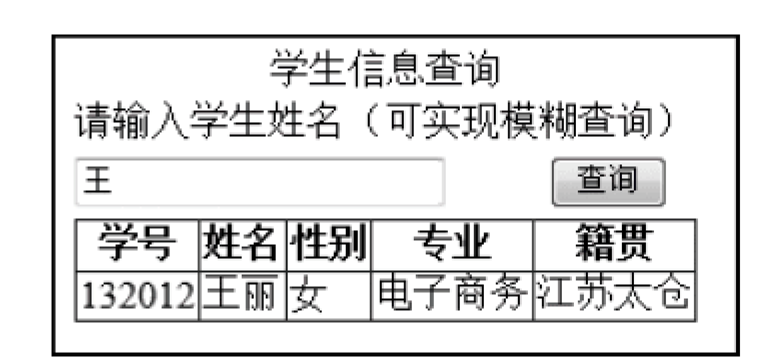


图 5.7　查询学生信息

【步骤 1】数据访问层定义方法,实现数据查询功能,代码如下所示:

```
public List<tbStuInfo> GetStuBySname(string name)
    {
        string sql=string.Format("select * from
        tbStuInfos where Sname like '%{0}%'", name);
        SqlDataReader dr = DBHelper. ExecuteReader (DBHelper. ConnectionString,
        CommandType.Text, sql);
        List<tbStuInfo> list=new List<tbStuInfo>();
        while(dr.Read())
        {
            tbStuInfo tbf=new tbStuInfo();
            tbf.Sno=Convert.ToString(dr["Sno"]);
            tbf.Sname=Convert.ToString(dr["Sname"]);
            tbf.Sex=Convert.ToString(dr["Sex"]);
            tbf.ZhuanYe=Convert.ToString(dr["ZhuanYe"]);
            tbf.JiGuan=Convert.ToString(dr["JiGuan"]);
            list.Add(tbf);
        }
        dr.Close();
        return list;
    }
```

【步骤 2】业务逻辑层编写代码如下所示:

```
public List<tbStuInfo> GetStuBySname(string name)
    {
        return new tbStuInfoService().GetStuBySname(name);
    }
```

【步骤 3】右击 StuMWeb，添加 StuCX.aspx 页面，添加一个文本框(TextBox)，一个按钮(Button)，一个数据绑定控件(GridView)，页面如图 5.8 所示，源视图代码如下所示：

```
<%@ Page Language="C#" AutoEventWireup="true" CodeFile="StuCX.aspx.cs" Inherits="StuCX"
%>
<!DOCTYPE html>
<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
<meta http-equiv="Content-Type" content="text/html; charset=utf-8"/>
    <title></title>
</head>
<body>
    <form id="form1" runat="server">
    <div>
    <table><tr><td colspan="2">学生信息查询</td></tr>
        <tr><td colspan="2">请输入学生姓名(可实现模糊查询)</td></tr>
         <tr><td><asp:TextBox ID="TextBox2" runat="server"></asp:TextBox></td><td>
             <asp:Button ID="btnCX" runat="server" Text="查询" />
            </td></tr>
         <tr><td colspan="2">
             <asp:GridView ID="GridView1" runat="server"></asp:GridView>
         </td></tr>
    </table>
    </div>
    </form>
</body>
</html>
```

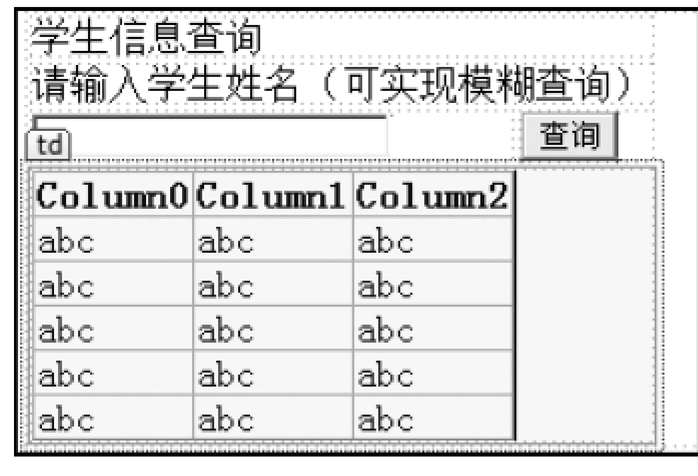



图 5.8　学生信息查询页面

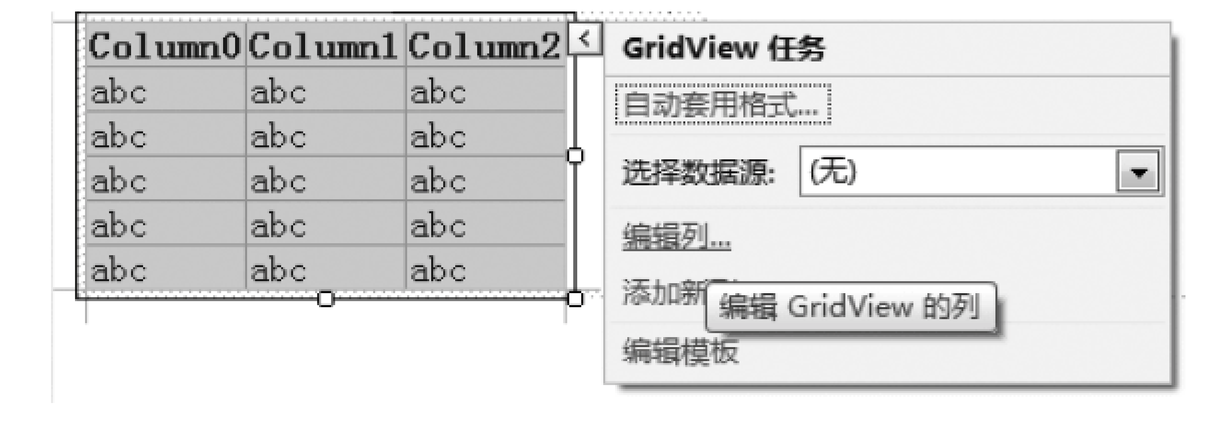


图 5.9　选中【编辑列...】命令

【步骤 4】选中 GridView，单击右上角的方块，弹出如图 5.9 所示的窗口，选中编辑列命令，弹出如图 5.10 所示的对话框。

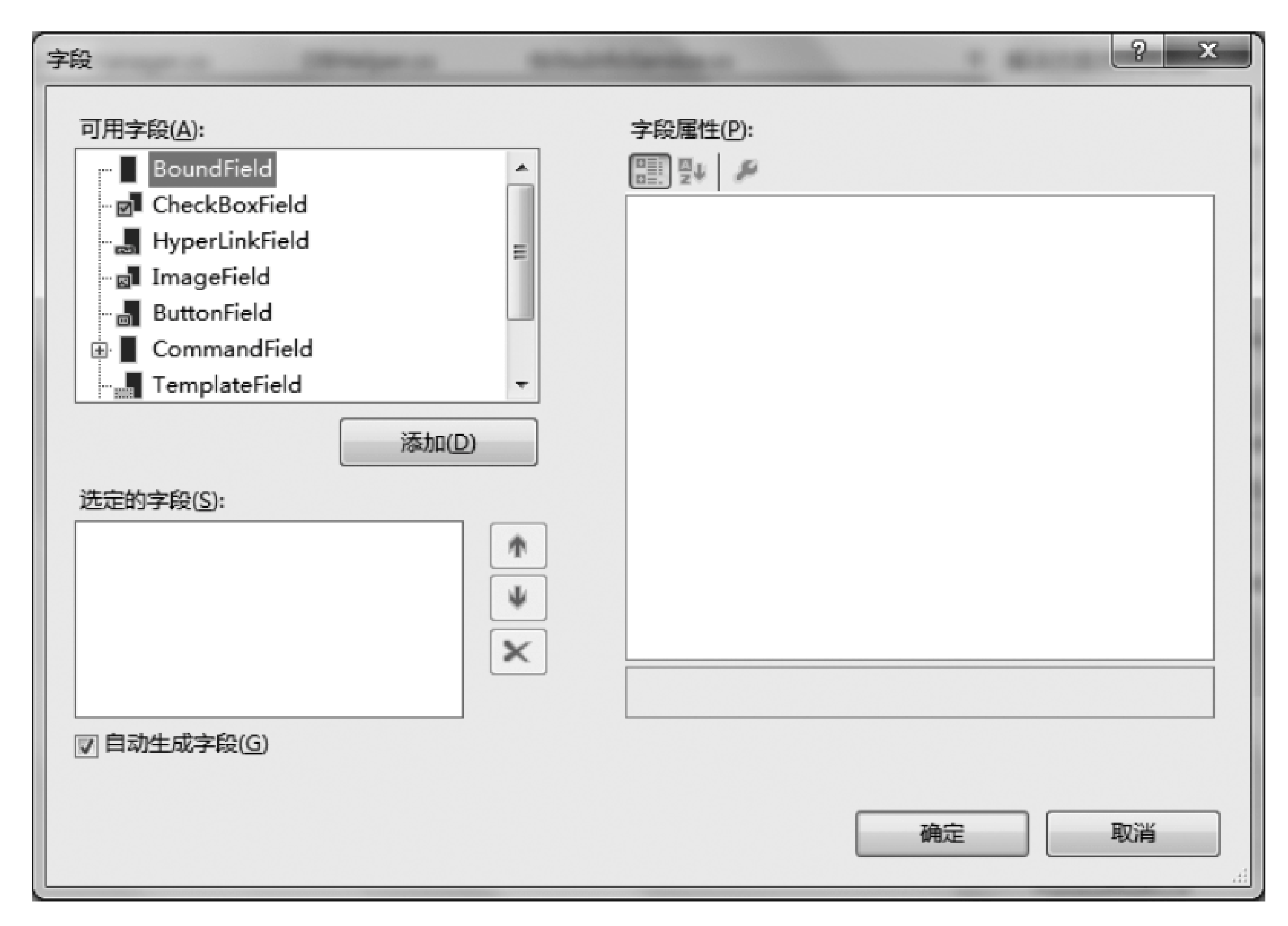


图 5.10 【字段】对话框

【步骤 5】添加 BoundField 字段，在 DataField 属性中设置显示的字段，HeaderText 显示列标题，如图 5.11 所示。这时会发现系统又为我们重复显示了一遍数据，如图 5.12 所示，解决方法是将 GridView 控件的 AutoGenerateColumns 属性设置为 False 即可。

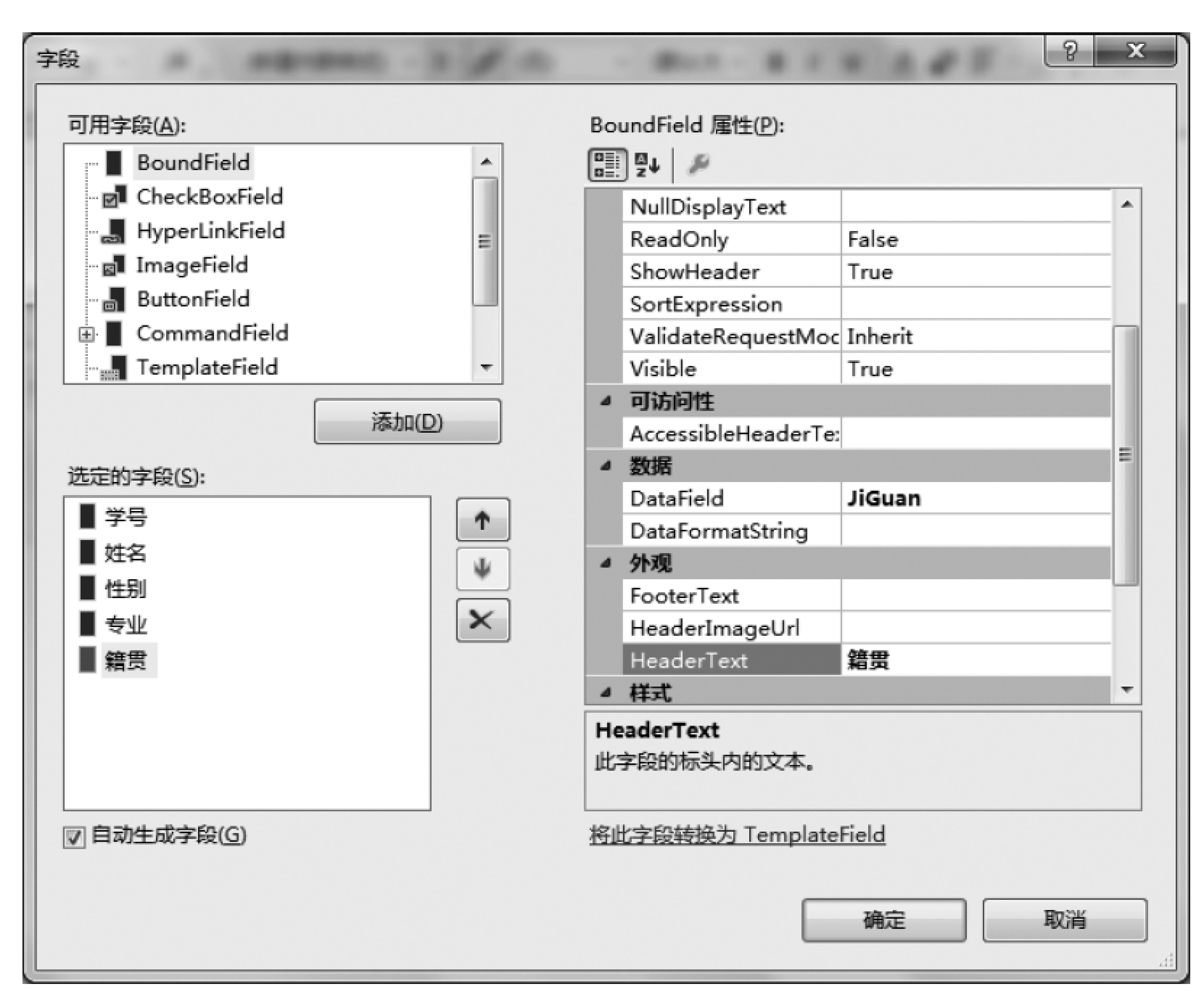


图 5.11 添加 BoundField 字段

AutoGenerateColumns 属性表示是否为数据源中的每一字段自动创建 BoundColumn 对象并在 GridView 控件中显示这些对象。

学生信息查询
请输入学生姓名（可实现模糊查询）
查询
asp:GridView#GridView1

学号	姓名	性别	专业	籍贯	Column0	Column1	Column2
数据绑定	数据绑定	数据绑定	数据绑定	数据绑定	abc	abc	abc
数据绑定	数据绑定	数据绑定	数据绑定	数据绑定	abc	abc	abc
数据绑定	数据绑定	数据绑定	数据绑定	数据绑定	abc	abc	abc
数据绑定	数据绑定	数据绑定	数据绑定	数据绑定	abc	abc	abc
数据绑定	数据绑定	数据绑定	数据绑定	数据绑定	abc	abc	abc

图 5.12　添加 BoundField 字段页面

【步骤 6】双击查询按钮，产生 Click 事件，编写代码如下：

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.UI;
using System.Web.UI.WebControls;
using StuBLL;
using StuModel;

public partial class StuCX : System.Web.UI.Page
{
    protected void Page_Load(object sender, EventArgs e)
    {
    }
    protected void btnCX_Click(object sender, EventArgs e)
    {
      GridView1.DataSource=new tbStuInfoManager().GetStuBySname(TextBox2.Text);
        GridView1.DataBind();
    }
}
```

5.2.3　删除数据方法与编码

我们通过单击 GridView 控件中的删除按钮，利用编码删除后台数据库中的相关数据。

【示例 5.3】　添加删除列，实现删除功能，如图 5.13 所示。

【步骤 1】选中 GridView，单击右上角的方块，在弹出的窗口中选中【编辑列…】命令，弹出【字段】对话框，添加 TemplateField 字段，并设置 HeaderText 属性为删除，如图 5.14 所示。



图 5.13　删除学生信息

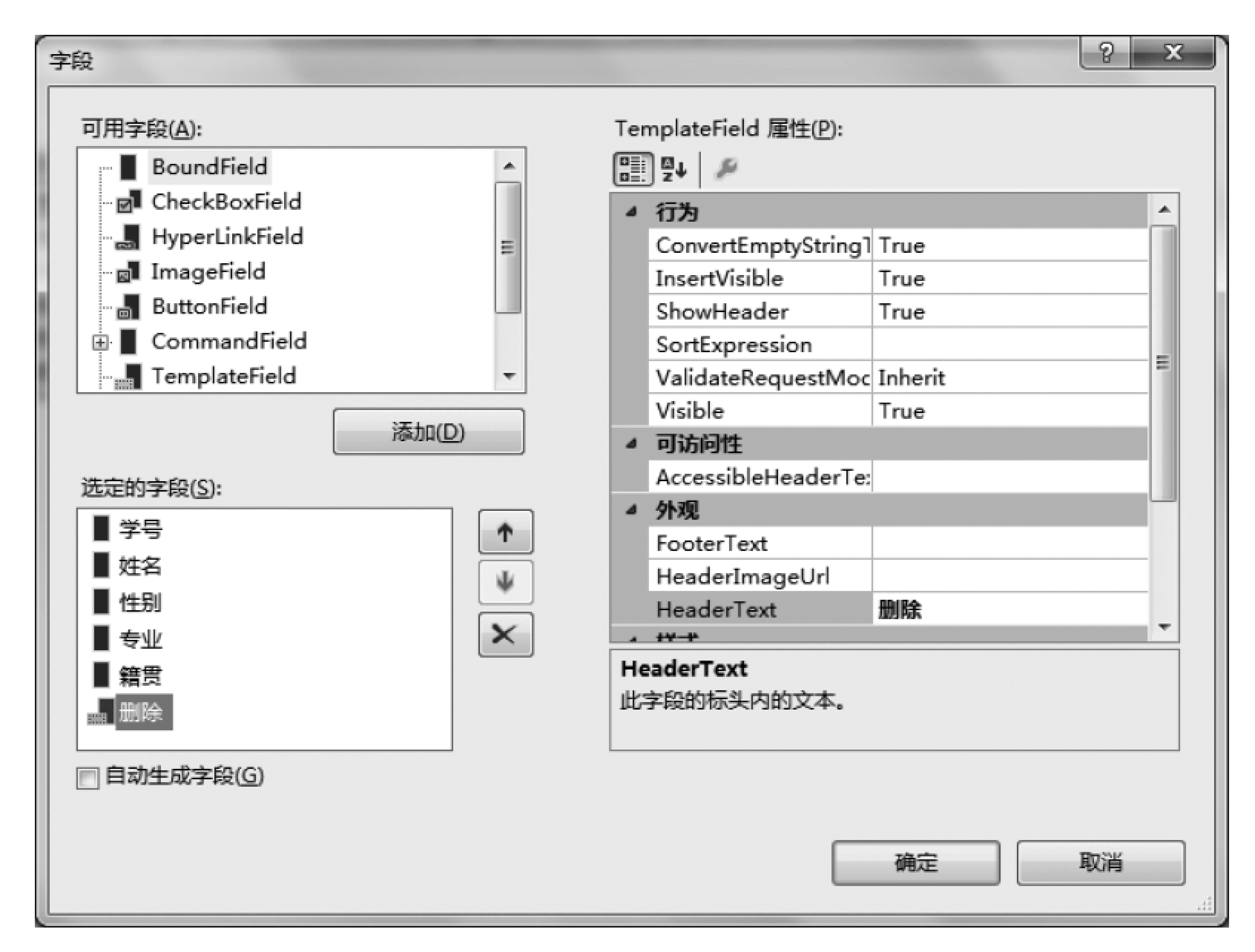


图 5.14　添加删除列

【步骤 2】再次选中 GridView，单击右上角的方块，在弹出的窗口中选中【编辑模板】命令，在 ItemTemplate 中拖曳 LinkButton 控件，并设置 CommandArgument 属性为'<%#Eval("Sno")%>'，CommandName 属性为"Delete"，关键代码如下所示：

```
<asp:TemplateField HeaderText="删除">
    <ItemTemplate>
  <asp:LinkButton ID="LinkButton1" runat="server" CommandName="Delete" CommandArgument
  ='<%#Eval("Sno")%>'>删除</asp:LinkButton>
    </ItemTemplate>
    </asp:TemplateField>
```

【步骤 3】数据访问层定义方法，实现删除学生信息功能。

```
public bool DeleteStuInfo(string sno)
      {
```

```
            string sql=string.Format("delete tbStuInfos where Sno='{0}'", sno);
            return DBHelper.ExecuteNonQuery(DBHelper.ConnectionString, CommandType.Text,
            sql)>0;
        }
```

【步骤 4】业务逻辑层代码如下所示：

```
public bool DeleteStuInfo(string sno)
        {
            return new tbStuInfoService().DeleteStuInfo(sno);
        }
```

【步骤 5】表示层代码如下所示：

```
protected void GridView1_RowDeleting(object sender, GridViewDeleteEventArgs e)
    {
        string sno = (GridView1.Rows[e.RowIndex].FindControl("LinkButton1") as
        LinkButton).CommandArgument;
        if(new tbStuInfoManager().DeleteStuInfo(sno))
        {
            Response.Write("<script>alert('删除成功')</script>");
          GridView1.DataSource=new tbStuInfoManager().GetStuBySname(TextBox2.Text);
            GridView1.DataBind();
        }
        else
            Response.Write("<script>alert('删除失败')</script>");
    }
```

5.2.4　修改数据方法与编码

GridView 能够实现基于单元格的更新功能。基本思路：先使要更新的行处于编辑模式，然后在该行上直接修改。修改后，将数据更新到数据库，最后重新绑定 GridView。

【示例 5.4】　添加编辑列，实现修改功能，如图 5.15 所示。



图 5.15　修改学生信息

【步骤 1】数据访问层编写方法，根据学生学号实现修改功能。

```
public bool ModifyStuInfo(tbStuInfo tbs)
        {
            string sql=string.Format("update tbStuInfos set Sname='{0}',Sex='{1}',ZhuanYe='
```

```
        {2}',JiGuan='{3}' where Sno='{4}'", tbs.Sname, tbs.Sex, tbs.ZhuanYe, tbs.
        JiGuan, tbs.Sno);
        return DBHelper.ExecuteNonQuery(DBHelper.ConnectionString, CommandType.Text,
        sql)>0;
    }
```

【步骤 2】业务逻辑层编写方法,调用数据访问层方法。

```
public bool ModifyStuInfo(tbStuInfo tbs)
        {
            return new tbStuInfoService().ModifyStuInfo(tbs);
        }
```

【步骤 3】设置模板列。选中 GridView,单击右上角的方块,在弹出的窗口中选中【编辑列...】命令,弹出【字段】对话框,选择学号列,ReadOnly 属性设置为"True"(学号不可修改),依次选择姓名、性别、专业、籍贯,单击右下角"将此字段转换为 TemplateField",如图 5.16 所示。

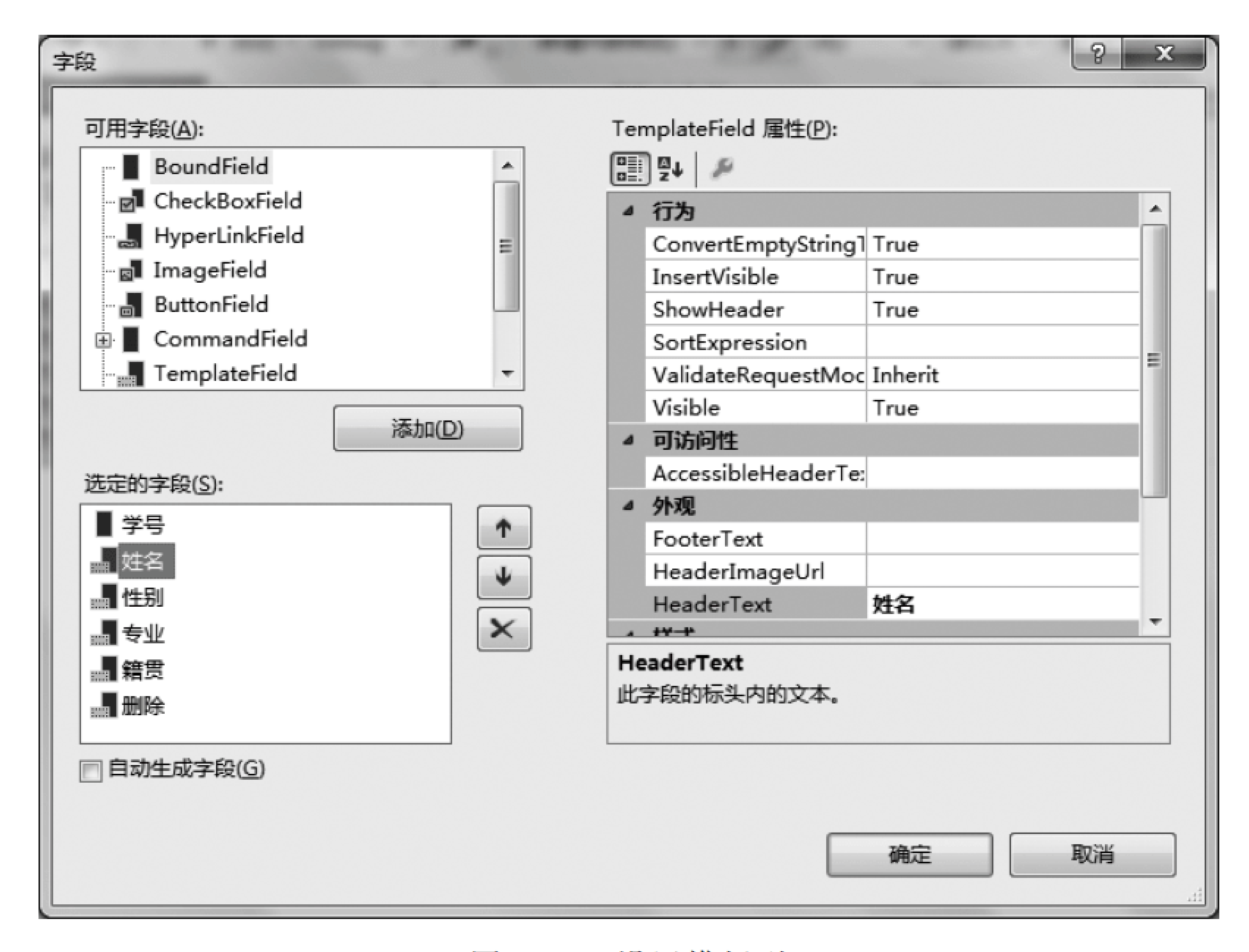


图 5.16 设置模板列

【步骤 4】添加命令按钮。添加 CommandField,它提供了在数据绑定控件中执行选择、编辑、插入或删除操作,设置 HeaderText 属性为"操作",ShowEditButton 属性为"True",如图 5.17 所示。

【步骤 5】编写 RowEditing 处理事件。RowEditing 在 GridView 控件进入编辑模式之前被引发。

```
protected void GridView1_RowEditing(object sender, GridViewEditEventArgs e)
```

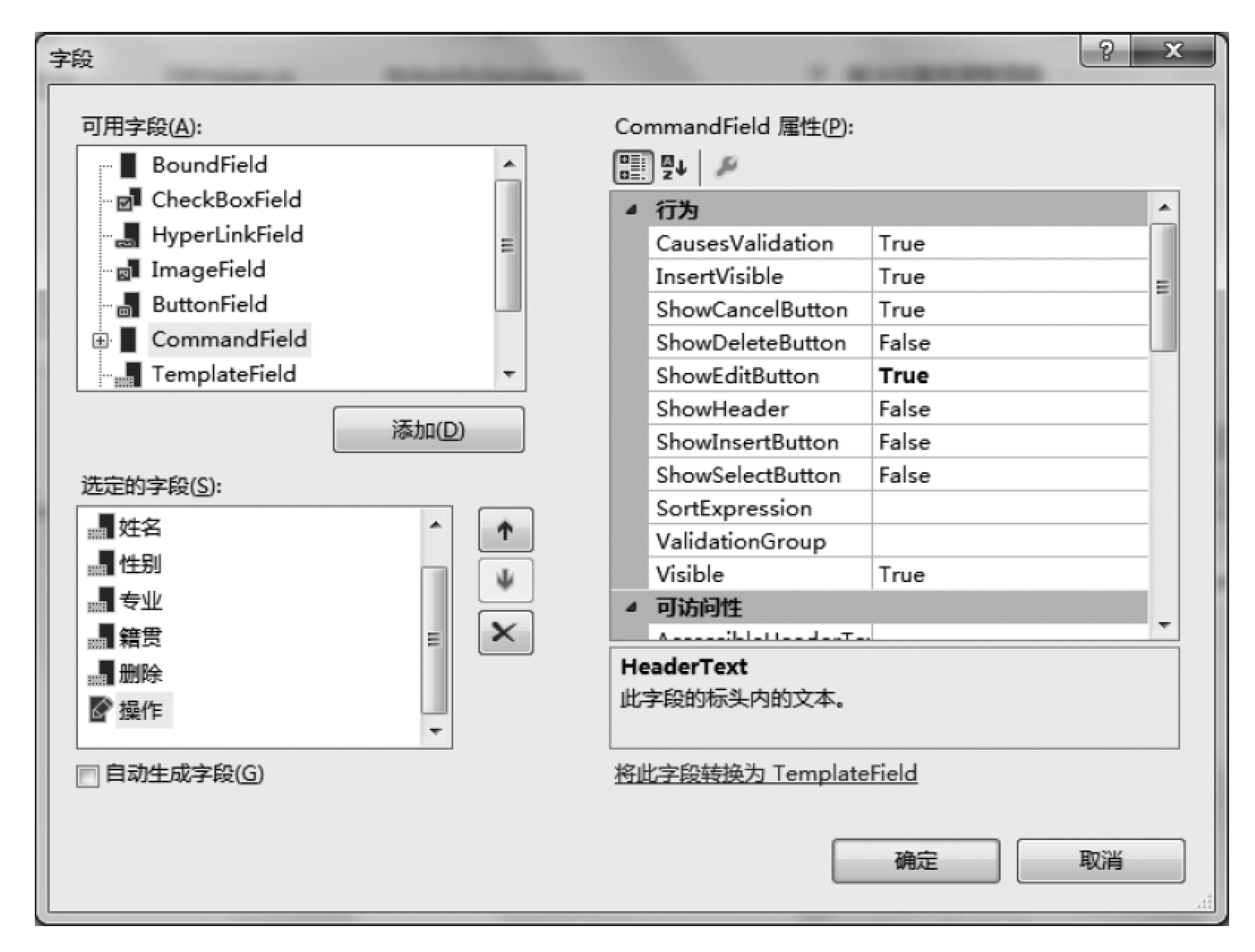


图 5.17　添加编辑列

```
    {
        GridView1.EditIndex=e.NewEditIndex;
                                //EditIndex 可以获取或设置要编辑的行的索引。当属性被设置为
                                //某一行的索引,该行进入编辑状态。
      GridView1.DataSource=new tbStuInfoManager().GetStuBySname(TextBox2.Text);
      GridView1.DataBind();
    }
```

【步骤 6】编写 RowUpdating 处理事件。RowUpdating 在单击某一行的“更新”按钮以后,在 GridView 控件对该行进行更新之前被引发。

```
protected void GridView1_RowUpdating(object sender, GridViewUpdateEventArgs e)
    {
        tbStuInfo tbstu=new tbStuInfo();
        tbstu.Sno= (GridView1. Rows [e. RowIndex]. FindControl ( " LinkButton1 ") as
        LinkButton).CommandArgument;
        tbstu.Sname= (GridView1.Rows[e.RowIndex].FindControl("TextBox1") as TextBox).
        Text;
        tbstu.Sex= (GridView1.Rows[e.RowIndex].FindControl("TextBox2")as TextBox).Text;
        tbstu.ZhuanYe= (GridView1.Rows[e.RowIndex].FindControl("TextBox3") as TextBox).
        Text;
        tbstu.JiGuan= (GridView1.Rows[e.RowIndex].FindControl("TextBox4") as TextBox).
        Text;
        if(new tbStuInfoManager().ModifyStuInfo(tbstu))
        {
```

```
            Response.Write("<script>alert('修改成功')</script>");
            GridView1.EditIndex=-1;              //退出编辑状态
        GridView1.DataSource=new tbStuInfoManager().GetStuBySname(TextBox2.Text);
        GridView1.DataBind();
        }
        else
            Response.Write("<script>alert('修改失败')</script>");
    }
```

【步骤 7】编写 RowCancelingEdit 处理事件。RowCancelingEdit 在单击"取消"按钮之后,在该行退出编辑模式之前被引发。

```
protected void GridView1_RowCancelingEdit(object sender,
GridViewCancelEditEventArgs e)
    {
    GridView1.EditIndex=-1;                   //退出编辑状态
    GridView1.DataSource=new tbStuInfoManager().GetStuBySname(TextBox2.Text);
    GridView1.DataBind();
    }
```

5.3　项 目 训 练

通过对以上内容的学习,了解了对数据进行增、删、改、查的原理,同时了解了相关控件属性的设置方法,现在我们回到项目导入的任务中来。

【步骤 1】搭建系统框架,添加各层之间的依赖关系,如图 5.18 所示。

【步骤 2】创建数据库 db_Sell,添加数据表 tb_Companys,如图 5.19 所示,并配置 web.config。



图 5.18　搭建系统框架

OEM-20130723TNF....- dbo.tb_Companys

列名	数据类型	允许 Null 值
Id	int	☐
CompanyName	varchar(50)	☑
CompanyAddress	varchar(50)	☑
Postalcode	varchar(50)	☑
Tel	varchar(50)	☑
Fax	varchar(50)	☑
Linkman	varchar(50)	☑
Email	varchar(50)	☑

图 5.19　数据表 tb_Companys

```
<connectionStrings>
  <add name="DefaultConnection" providerName="System.Data.SqlClient" connectionString
  ="Data Source=.;Initial Catalog=db_Sell;Integrated Security=True" />
</connectionStrings>
```

【步骤 3】根据数据表，在实体层添加类，编写代码如下：

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace CompanyInfoModel
{
   public class Company
     {
         private int id;
         public int Id
         {
             get { return id; }
             set { id=value; }
         }
         private string companyName;
         public string CompanyName
         {
             get { return companyName; }
             set { companyName=value; }
         }
         private string companyAddress;
         public string CompanyAddress
         {
             get { return companyAddress; }
             set { companyAddress=value; }
         }
         private string postalcode;
         public string Postalcode
         {
             get { return postalcode; }
             set { postalcode=value; }
         }
         private string tel;
         public string Tel
         {
             get { return tel; }
```

```
            set { tel=value; }
        }
        private string fax;
        public string Fax
        {
            get { return fax; }
            set { fax=value; }
        }
        private string linkman;
        public string Linkman
        {
            get { return linkman; }
            set { linkman=value; }
        }
        private string email;
        public string Email
        {
            get { return email; }
            set { email=value; }
        }
    }
}
```

【步骤 4】添加数据访问类 DBHelper.cs,并编写代码,见第 3 章内容。

【步骤 5】数据访问层添加类,编写代码如下:

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using CompanyInfoModel;
using System.Data;
using System.Data.SqlClient;

namespace CompanyInfoDAL
{
    public  class CompanyService
    {
//公司信息一览表
        public List<Company> GetAllCompanys()
        {
            List<Company>  list=new List<Company>();
            string sql="select * from dbo.tb_Companys";
            SqlDataReader dr=DBHelper.ExecuteReader(DBHelper.
```

```
                ConnectionString, CommandType.Text, sql);
                while(dr.Read())
                {
                    Company compan=new Company();
                    compan.CompanyAddress=Convert.ToString(dr["CompanyAddress"]);
                    compan.CompanyName=Convert.ToString(dr["CompanyName"]);
                    compan.Email=Convert.ToString(dr["Email"]);
                    compan.Fax=Convert.ToString(dr["Fax"]);
                    compan.Id=Convert.ToInt32(dr["Id"]);
                    compan.Linkman=Convert.ToString(dr["Linkman"]);
                    compan.Postalcode=Convert.ToString(dr["Postalcode"]);
                    compan.Tel=Convert.ToString(dr["Tel"]);
                    list.Add(compan);
                }
                dr.Close();
                return list;
            }
//添加公司信息
            public bool AddCompany(Company company)
            {
                string sql= string.Format("insert into dbo.tb_Companys(CompanyAddress,
                CompanyName,Email,Fax,Linkman,Postalcode,Tel)values('{0}','{1}','{2}','{3}
                ','{4}','{5}','{6}')", company.CompanyAddress, company.CompanyName, company.
                Email, company.Fax, company.Linkman, company.Postalcode, company.Tel);
                return DBHelper.ExecuteNonQuery(DBHelper.ConnectionString,
                CommandType.Text, sql)>0;
            }
//删除公司信息
            public bool DeleteCompany(int id)
            {
                string sql=string.Format("delete dbo.tb_Companys where Id='{0}'", id);
                return DBHelper. ExecuteNonQuery (DBHelper. ConnectionString, CommandType.
                Text, sql)>0;
            }
//修改公司信息
            public bool ModifyCompany(Company company)
            {
                string sql= string.Format("update dbo.tb_Companys set CompanyName= '{0}',
                Email='{1}',Linkman='{2}',Tel='{3}' where Id='{4}'", company.CompanyName,
                company.Email, company.Linkman, company.Tel, company.Id);
                return DBHelper. ExecuteNonQuery (DBHelper. ConnectionString, CommandType.
                Text, sql)>0;
            }
        }
```

```
}
```

【步骤 6】业务逻辑层添加类，编写代码如下：

```
using CompanyInfoModel;
using CompanyInfoDAL;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
namespace CompanyInfoBLL
{
   public class CompanyManager
    {
       public List<Company> GetAllCompanys()
       {
           return new CompanyService().GetAllCompanys();
       }
       public bool AddCompany(Company company)
       {
           return new CompanyService().AddCompany(company);
       }
       public bool DeleteCompany(int id)
       {
           return new CompanyService().DeleteCompany(id);
       }
       public bool ModifyCompany(Company company)
       {
           return new CompanyService().ModifyCompany(company);
       }
    }
}
```

【步骤 7】制作“添加公司信息”界面，如图 5.1 所示，源视图代码如下：

```
<table style="height: 376px;border:1px; text-align:center">
             <tr>
        <td colspan="2" style="height: 39px; color: white; background-color: #003399;">
           添加公司信息</td>
             </tr>
             <tr>
                  <td style="width: 100px;text-align:center">
                      公司名称:</td>
                  <td style="width: 334px" >
<asp:TextBox ID="txtCompanyName" runat="server" Width="280px"></asp:TextBox>
```

```
</td>
                    </tr>
                    <tr>
                        <td style="text-align:center"  >
                            地 址:</td>
                        <td>
                            <asp:TextBox ID="txtCompanyAddress" runat="server"  Width="280px"
                            ></asp:TextBox></td>
                    </tr>
                    <tr>
                        <td  style="text-align:center">
                            邮政编码:</td>
                        <td>
                            <asp:TextBox ID="txtPostalcode" runat="server"  Width= "280px"></
                            asp:TextBox></td>
                    </tr>
                    <tr>
                        <td  style="text-align:center"  >
                            联系电话:</td>
                        <td>
                            <asp:TextBox ID="txtTel" runat="server"  Width="280px">
                            </asp:TextBox></td>
                    </tr>
                    <tr>
                        <td  style="text-align:center">
                            传真:</td>
                        <td>
                            <asp:TextBox ID="txtFax" runat="server"  Width="280px">
                            </asp:TextBox></td>
                    </tr>
                    <tr>
                        <td style="text-align:center">
                            联系人:</td>
                        <td>
                            <asp:TextBox ID="txtLinkman" runat="server"  Width="280px">
                            </asp:TextBox></td>
                    </tr>
                    <tr>
                        <td  style="text-align:center" class="auto-style1">
                            E-mail:</td>
                        <td class="auto-style1">
                            <asp:TextBox ID="txtEmail" runat="server"  Width="280px"></asp:
                            TextBox></td>
                    </tr>
```

```
            <tr>
                <td  colspan="2">
                    <asp:Button ID="Button1" runat="server" Text=" 添加 " OnClick="
                    Button1_Click" />
                </td>
            </tr>
        </table>
```

【步骤 8】“添加公司信息”界面后台代码如下所示：

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.UI;
using System.Web.UI.WebControls;
using CompanyInfoBLL;
using CompanyInfoModel;

public partial class AddCompany : System.Web.UI.Page
{
    protected void Page_Load(object sender, EventArgs e)
    {
    }
    protected void Button1_Click(object sender, EventArgs e)
    {
        Company company=new Company();
        company.CompanyName=txtCompanyName.Text;
        company.CompanyAddress=txtCompanyAddress.Text;
        company.Email=txtEmail.Text;
        company.Fax=txtFax.Text;
        company.Linkman=txtLinkman.Text;
        company.Postalcode=txtPostalcode.Text;
        company.Tel=txtTel.Text;
        if(new CompanyManager().AddCompany(company))
          Response.Write("<script>alert('添加成功')</script>");
        else
        Response.Write("<script>alert('添加失败')</script>");
    }
}
```

【步骤 9】制作“公司信息一览表”界面，如图 5.2 所示。

GridView 控件代码：

```
<asp:GridView ID="GridView1" runat="server" AutoGenerateColumns="False">
            <Columns>
```

```
        <asp:BoundField DataField="CompanyName" HeaderText="公司名称" >
          <ItemStyle Width="200px" HorizontalAlign="Center" />
          </asp:BoundField>
          <asp:BoundField DataField="Email" HeaderText="Email" >
          <ItemStyle Width="140px"  HorizontalAlign="Center" />
          </asp:BoundField>
          <asp:BoundField DataField="Linkman" HeaderText="联系人" >
          <ItemStyle Width="80px"  HorizontalAlign="Center"/>
          </asp:BoundField>
          <asp:BoundField DataField="Tel" HeaderText="电话" >
          <ItemStyle Width="140px"  HorizontalAlign="Center"/>
          </asp:BoundField>
       </Columns>
    </asp:GridView>
```

关键代码如下所示：

```
protected void Page_Load(object sender, EventArgs e)
    {
        if(!IsPostBack)
        {
            GridView1.DataSource=new CompanyManager().GetAllCompanys();
            GridView1.DataBind();
        }
    }
```

【步骤 10】设置“公司信息一览表”界面中 GridView 控件的删除列，如图 5.2 所示。GridView 控件添加代码：

```
<asp:TemplateField HeaderText="删除">
                    <ItemTemplate>
                        <asp:LinkButton ID="LinkButton1" runat="server" CommandName="
                        Delete" CommandArgument='<%#Eval("Id")%>'>删除</asp:LinkButton
                        >
                    </ItemTemplate>
                </asp:TemplateField>
```

关键代码：

```
protected void GridView1_RowDeleting(object sender, GridViewDeleteEventArgs e)
    {
        int id=Convert.ToInt32((GridView1.Rows[e.RowIndex].FindControl
        ("LinkButton1")as LinkButton).CommandArgument);
        if(new CompanyManager().DeleteCompany(id))
        {
            Response.Write("<script>alert('删除成功')</script>");
        GridView1.DataSource=new CompanyManager().GetAllCompanys();
```

```
            GridView1.DataBind();
            }
            else
                Response.Write("<script>alert('删除失败')</script>");
        }
```

【步骤 11】修改公司信息。将公司名称、Email、联系人、电话转换成模板,并添加编辑列,GridView 代码如下所示:

```
<asp:GridView ID="GridView1" runat="server" AutoGenerateColumns="False" OnRowDeleting
="GridView1_RowDeleting">
            <Columns>
                <asp:TemplateField HeaderText="公司名称">
                    <EditItemTemplate>
                        <asp:TextBox ID="txtCompanyName" runat="server" Text= '<%#
                        Bind("CompanyName")%>'></asp:TextBox>
                    </EditItemTemplate>
                    <ItemTemplate>
                        <asp:Label ID="Label1" runat="server" Text='<%# Bind
                        ("CompanyName")%>'></asp:Label>
                    </ItemTemplate>
                    <ItemStyle HorizontalAlign="Center" Width="200px" />
                </asp:TemplateField>
                <asp:TemplateField HeaderText="Email">
                    <EditItemTemplate>
                        <asp:TextBox ID="txtEmail" runat="server" Text='<%#  Bind("
                        Email")%>'></asp:TextBox>
                    </EditItemTemplate>
                    <ItemTemplate>
                        <asp:Label ID="Label2" runat="server" Text='<%# Bind
                        ("Email")%>'></asp:Label>
                    </ItemTemplate>
                    <ItemStyle HorizontalAlign="Center" Width="140px" />
                </asp:TemplateField>
                <asp:TemplateField HeaderText="联系人">
                    <EditItemTemplate>
                        <asp:TextBox ID="txtLinkName" runat="server" Text='<% #Bind("
                        Linkman")%>'></asp:TextBox>
                    </EditItemTemplate>
                    <ItemTemplate>
                        <asp:Label ID="Label3" runat="server" Text='<%# Bind
                        ("Linkman")%>'></asp:Label>
                    </ItemTemplate>
                    <ItemStyle HorizontalAlign="Center" Width="80px" />
                </asp:TemplateField>
```

```
                <asp:TemplateField HeaderText="电话">
                    <EditItemTemplate>
                        <asp:TextBox ID="txtTel" runat="server" Text='<%#Bind("
                        Tel")%>'></asp:TextBox>
                    </EditItemTemplate>
                    <ItemTemplate>
                        <asp:Label ID="Label4" runat="server" Text='<%#Bind("Tel")%
                       >'></asp:Label>
                    </ItemTemplate>
                    <ItemStyle HorizontalAlign="Center" Width="140px" />
                </asp:TemplateField>
                <asp:CommandField ShowEditButton="True" />
                <asp:TemplateField HeaderText="删除">
                    <ItemTemplate>
                        <asp:LinkButton ID="LinkButton1" runat="server" CommandName="
                        Delete" CommandArgument='<%#Eval("Id")%>'>删除</asp:LinkButton
                        >
                    </ItemTemplate>
                </asp:TemplateField>
            </Columns>
        </asp:GridView>
```

关键代码：

```
protected void GridView1_RowEditing(object sender, GridViewEditEventArgs e)
    {
        GridView1.EditIndex=e.NewEditIndex;
        GridView1.DataSource=new CompanyManager().GetAllCompanys();
        GridView1.DataBind();
    }
    protected void GridView1_RowUpdating(object sender, GridViewUpdateEventArgs e)
    {
        Company company=new Company();
        company.Id=Convert.ToInt32((GridView1.Rows[e.RowIndex].FindControl
        ("LinkButton1")as LinkButton).CommandArgument);
        company.CompanyName= (GridView1.Rows[e.RowIndex].FindControl
        ("txtCompanyName")as TextBox).Text;
        company.Linkman= (GridView1.Rows[e.RowIndex].FindControl
        ("txtLinkName")as TextBox).Text;
        company.Tel= (GridView1.Rows[e.RowIndex].FindControl("txtTel")as TextBox).Text;
        company.Email= (GridView1.Rows[e.RowIndex].FindControl("txtEmail")as TextBox).
        Text;
        if(new CompanyManager().ModifyCompany(company))
        {
            Response.Write("<script>alert('修改成功')</script>");
```

```
                GridView1.EditIndex=-1;
                GridView1.DataSource=new CompanyManager().GetAllCompanys();
                GridView1.DataBind();
            }
            else
                Response.Write("<script>alert('修改失败')</script>");
        }
        protected void GridView1_RowCancelingEdit(object sender,
        GridViewCancelEditEventArgs e)
        {
            GridView1.EditIndex=-1;
            GridView1.DataSource=new CompanyManager().GetAllCompanys();
            GridView1.DataBind();
        }
```

5.4 平行项目训练

1. 训练内容

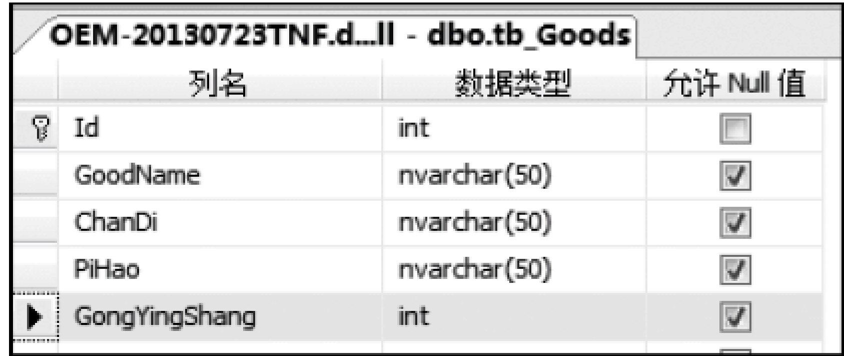
利用编码对数据进行修改操作，实现基于单元格的更新功能。

2. 训练目的

进一步训练和巩固学生对数据进行修改的编码能力，重点是外键的处理。

3. 训练过程

【步骤 1】在数据库中添加数据表 dbo. tb_Goods，GongYingShang 是外键，如图 5.20 所示。



OEM-20130723TNF.d...ll - dbo.tb_Goods

列名	数据类型	允许 Null 值
Id	int	☐
GoodName	nvarchar(50)	☑
ChanDi	nvarchar(50)	☑
PiHao	nvarchar(50)	☑
GongYingShang	int	☑

图 5.20 商品表

【步骤 2】实体层添加类 Good. cs，编写代码如下所示：

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace CompanyInfoModel
{
   public class Good
    {
        private int id;
        public int Id
        {
            get { return id; }
```

```
            set { id=value; }
        }
        private string goodName;
        public string GoodName
        {
            get { return goodName; }
            set { goodName=value; }
        }
        private string chanDi;
        public string ChanDi
        {
            get { return chanDi; }
            set { chanDi=value; }
        }
        private string piHao;
        public string PiHao
        {
            get { return piHao; }
            set { piHao=value; }
        }
        private Company company;
        public Company Company                    //外键
        {
            get { return company; }
            set { company=value; }
        }
    }
}
```

【步骤 3】数据访问层编写代码如下所示：

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using CompanyInfoModel;
using System.Data;
using System.Data.SqlClient;

namespace CompanyInfoDAL
{
   public class GoodService
    {
       public List<Good> GetAllGood()
```

```
        {
            string sql="select * from dbo.tb_Goods";
            SqlDataReader dr = DBHelper. ExecuteReader ( DBHelper. ConnectionString,
            CommandType.Text, sql);
            List<Good> list=new List<Good> ();
            while(dr.Read())
            {
                Good gd=new Good();
                gd.Id=Convert.ToInt32(dr["Id"]);
                gd.GoodName=Convert.ToString(dr["GoodName"]);
                gd.ChanDi=Convert.ToString(dr["ChanDi"]);
                gd.PiHao=Convert.ToString(dr["PiHao"]);
                gd.Company=new CompanyService().GetCompanyById(Convert.ToInt32(dr["
                GongYingShang"]));
//根据外键 GongYingShang 获取一条记录
                list.Add(gd);
            }
            dr.Close();
            return list;
        }
        public bool ModifyGood(Good gd)
        {
            string sql=string.Format("update dbo.tb_Goods set GoodName='{0}',ChanDi='{1}
            ',PiHao='{2}',GongYingShang='{3}' where Id='{4}'", gd.GoodName, gd.ChanDi,
            gd.PiHao, gd.Company.Id, gd.Id);
            return DBHelper.ExecuteNonQuery(DBHelper.ConnectionString, CommandType.Text,
            sql)>0;
        }
    }
}
```

【步骤 4】业务逻辑层编写代码如下所示：

```
using CompanyInfoModel;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using CompanyInfoDAL;

namespace CompanyInfoBLL
{
   public class GoodManager
    {
```

```
        public List<Good> GetAllGood()
        {
            return new GoodService().GetAllGood();
        }
        public bool ModifyGood(Good gd)
        {
            return new GoodService().ModifyGood
            (gd);
        }
    }
}
```

商品信息管理

商品名称	产地	批号	供应商
数据绑定	数据绑定	数据绑定	数据绑定
数据绑定	数据绑定	数据绑定	数据绑定
数据绑定	数据绑定	数据绑定	数据绑定
数据绑定	数据绑定	数据绑定	数据绑定
数据绑定	数据绑定	数据绑定	数据绑定

图 5.21　商品信息管理界面设计

【步骤 5】制作“商品信息管理”界面，如图 5.21 所示，源视图代码如下所示。

```
<asp:GridView ID="GridView1" runat="server" AutoGenerateColumns="False">
            <Columns>
                <asp:BoundField DataField="GoodName" HeaderText="商品名称" />
                <asp:BoundField DataField="ChanDi" HeaderText="产地" />
                <asp:BoundField DataField="PiHao" HeaderText="批号" />
                <asp:BoundField DataField="Company" HeaderText="供应商" />
            </Columns>
        </asp:GridView>
```

后台代码如下所示：

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.UI;
using System.Web.UI.WebControls;
using CompanyInfoBLL;
using CompanyInfoModel;

public partial class GoodList : System.Web.UI.Page
{
    protected void Page_Load(object sender, EventArgs e)
    {
        if(!IsPostBack)
        {
            GridView1.DataSource=new GoodManager().GetAllGood();
            GridView1.DataBind();
        }
    }
}
```

【步骤 6】浏览“商品信息管理”界面，如图 5.22 所示。

【步骤 7】将供应商列转换成模板，页图浏览如图 5.23 所示。修改代码如下：

商品信息管理			
商品名称	产地	批号	供应商
添加剂	江苏太仓	s12321	CompanyInfoModel.Company

图 5.22　商品信息管理页面一

商品信息管理			
商品名称	产地	批号	供应商
添加剂	江苏太仓	s12321	奇奇科技有限公司

图 5.23　商品信息管理页面二

```
<asp:TemplateField HeaderText="供应商">
                  <EditItemTemplate>
                      <asp:TextBox ID="TextBox1" runat="server" Text='<%#Bind
                      ("Company.CompanyName")%>'></asp:TextBox>
                  </EditItemTemplate>
                  <ItemTemplate>
                      <asp:Label ID="Label1" runat="server" Text='<%#Bind
                      ("Company.CompanyName")%>'></asp:Label>
                  </ItemTemplate>
              </asp:TemplateField>
```

【步骤 8】添加编辑列，将商品名称、产地、批号转换成模板，代码如下所示：

```
<asp:GridView ID="GridView1" runat="server" AutoGenerateColumns="False">
            <Columns>
                <asp:TemplateField HeaderText="商品名称">
                    <EditItemTemplate>
                        <asp:TextBox ID="txtName" runat="server" Text='<%#Bind("
                        GoodName")%>'></asp:TextBox>
                    </EditItemTemplate>
                    <ItemTemplate>
                        <asp:Label ID="Label2" runat="server" Text='<%#Bind
                        ("GoodName")%>'></asp:Label>
                    </ItemTemplate>
                </asp:TemplateField>
                <asp:TemplateField HeaderText="产地">
                    <EditItemTemplate>
                        <asp:TextBox ID="txtChanDi" runat="server" Text='<%#  Bind("
                        ChanDi")%>'></asp:TextBox>
                    </EditItemTemplate>
                    <ItemTemplate>
                        <asp:Label ID="Label3" runat="server" Text='<%#Bind
                        ("ChanDi")%>'></asp:Label>
                    </ItemTemplate>
                </asp:TemplateField>
                <asp:TemplateField HeaderText="批号">
                    <EditItemTemplate>
```

```
            <asp:TextBox ID="txtPiHao" runat="server" Text='<%#Bind("
            PiHao")%>'></asp:TextBox>
        </EditItemTemplate>
        <ItemTemplate>
            <asp:Label ID="Label4" runat="server" Text='<%#Bind
            ("PiHao")%>'></asp:Label>
        </ItemTemplate>
    </asp:TemplateField>
    <asp:TemplateField HeaderText="供应商">
        <EditItemTemplate>
            <asp:DropDownList ID="DropDownList1" runat="server">
            </asp:DropDownList>
        </EditItemTemplate>
        <ItemTemplate>
            <asp:Label ID="Label1" runat="server" Text='<%#Bind
            ("Company.CompanyName")%>'></asp:Label>
        </ItemTemplate>
    </asp:TemplateField>
</Columns>
</asp:GridView>
```

说明：商品名称、产地和批号列期望用户手动输入信息，因此用文本框控件，而供应商是外键，为避免用户的错误输入，使用 DropDownList 下拉列表框控件供用户选择，下拉列表框的数据根据数据库动态更新，所以在后台代码中动态绑定。

【步骤 9】获取主键值。设置 GridView 控件的 DataKeyNames 属性为 Id，如图 5.24 所示，获取主键的值的代码如下所示：

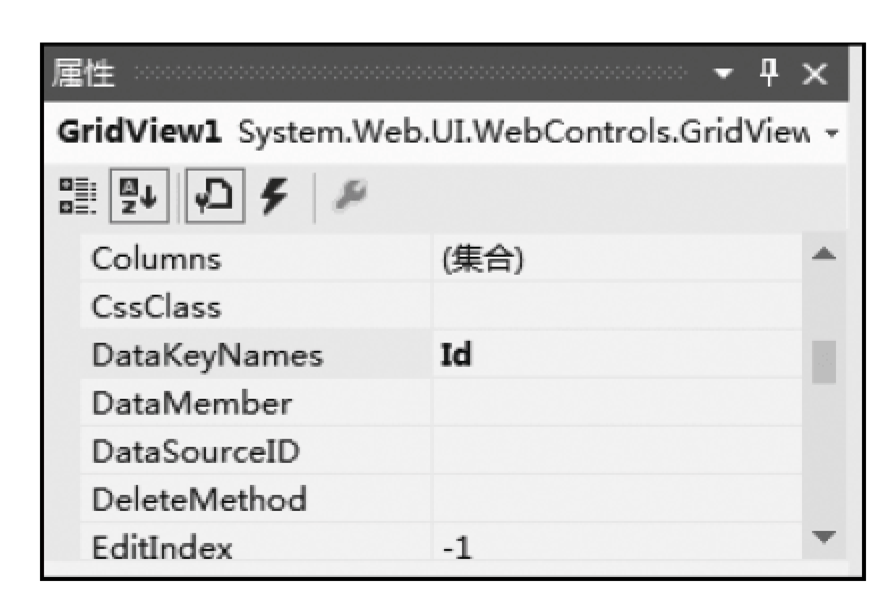


图 5.24　设置 GridView 控件的 DataKeyNames 属性

```
Convert.ToInt32(GridView1.DataKeys[index].Value)
```

【步骤 10】后台代码如下所示：

```
protected void GridView1_RowEditing(object sender, GridViewEditEventArgs e)
    {
        GridView1.EditIndex=e.NewEditIndex;
        GridView1.DataSource=new GoodManager().GetAllGood();
        GridView1.DataBind();
        //设置供应商列信息
        DropDownList ddl=GridView1.Rows[e.NewEditIndex].FindControl
        ("DropDownList1")as DropDownList;
        ddl.DataSource=new CompanyManager().GetAllCompanys();
        ddl.DataTextField="CompanyName";              //用于显示的字段
        ddl.DataValueField="Id";                      //用于存值的字段
        ddl.DataBind();
    }
    protected void GridView1_RowUpdating(object sender, GridViewUpdateEventA-
    rgs e)
    {
        Good gd=new Good();
        gd.Id=Convert.ToInt32(GridView1.DataKeys[e.RowIndex].Value);
        gd.GoodName=(GridView1.Rows[e.RowIndex].FindControl("txtName")as TextBox).
        Text;
        gd.ChanDi=(GridView1.Rows[e.RowIndex].FindControl("txtChanDi")as TextBox).
        Text;
        gd.PiHao=(GridView1.Rows[e.RowIndex].FindControl("txtPiHao")as TextBox).Text;
        int n=Convert.ToInt32((GridView1.Rows[e.RowIndex].FindControl("DropDownList1")
        as DropDownList).SelectedValue);
        gd.Company=new CompanyManager().GetCompanyById(n);
        if(new GoodManager().ModifyGood(gd))
        { Response.Write("<script>alert('修改成功')</script>");
        GridView1.EditIndex=-1;
        GridView1.DataSource=new GoodManager().GetAllGood();
        GridView1.DataBind();
        }
        else
            Response.Write("<script>alert('修改失败')</script>");
    }
    protected void GridView1_RowCancelingEdit(object sender, GridViewCancelEditEve-
    ntArgs e)
    {
        GridView1.EditIndex=-1;
        GridView1.DataSource=new GoodManager().GetAllGood();
        GridView1.DataBind();
```

```
}
```

【步骤 11】运行界面如图 5.25 所示，单击编辑后界面如图 5.26 所示，思考什么地方需要修改一下？

商品信息管理				
商品名称	产地	批号	供应商	编辑
添加剂	江苏太仓	s12321	奇奇科技有限公司	编辑
冰红茶121	江苏昆山	s332211	新世纪集团	编辑

图 5.25 商品信息管理界面

商品信息管理				
商品名称	产地	批号	供应商	编辑
添加剂	江苏太仓	s12321	奇奇科技有限公司	编辑
冰红茶121	江苏昆山	s332211	新科技有限公司	更新 取消

图 5.26 修改商品信息界面

【步骤 12】怎么在编辑状态中默认显示编辑前的“供应商”呢？可以在 EditItemTemplate 中添加一个隐藏的控件 HiddenField，使用该控件的 Value 属性绑定供应商编号，这样在编辑状态中就可以找到该控件，取得 Value 属性值赋给 DropDownList 的 SelectValue 即可，添加的代码如下：

```
<EditItemTemplate>
                    <asp:HiddenField ID="HiddenField1" runat="server" Value=
                    '<%#Eval("Company.Id")%>' />
                        <asp:DropDownList ID="DropDownList1" runat="server">
                        </asp:DropDownList>
                    </EditItemTemplate>
```

在 GridView1_RowEditing 事件最后添加代码：

```
ddl.SelectedValue=(GridView1.Rows[e.NewEditIndex].FindControl
("HiddenField1")as HiddenField).Value;
```

运行结果如图 5.27 所示。

商品信息管理				
商品名称	产地	批号	供应商	编辑
添加剂	江苏太仓	s12321	奇奇科技有限公司	编辑
冰红茶121	江苏昆山	s332211	新世纪集团	更新 取消

图 5.27 修改商品信息界面

5.5 总 结

本章介绍了对数据进行增、删、改、查操作的原理，同时通过项目训练“公司信息管理”及平行项目“商品信息管理”整体训练了利用编码实现增、删、改、查操作的方法，还有外键的处理方法。

5.6 习 题

1. 简述实现 GridView 基于单元格更新的实现过程。
2. 为什么在 EditItemTemplate 中添加 HiddenField 控件？
3. 实现在线考试系统中的专业信息管理功能，包括专业更新、删除专业功能。数据表如图 5.28 和图 5.29 所示。数据显示如图 5.30 所示（用 GridView 控件来实现）。

PC201011030...o.tbZhuanYes

列名	数据类型	允许 Null 值
Id	int	☐
Name	nvarchar(50)	☑
Yid	int	☑
JoinTime	datetime	☑
		☐

图 5.28 专业表



PC201011030...o.tbYuanXis

列名	数据类型	允许 Null 值
Id	int	☐
Name	nvarchar(50)	☑
JoinTime	datetime	☑

图 5.29 院系表

专业名称	所属院系	加入时间		删除
计算机网络	软件与服务外包学院	2010-1-1	编辑	删除
电子商务	软件与服务外包学院	2015-5-20	编辑	删除
报关	港口系	2012-2-2	编辑	删除
软件技术	软件与服务外包学院	2010-1-1	编辑	删除
会计	港口系（软件与服务外包学院 / 港口系 / 中德学院121）	2012-1-6	更新 取消	删除

图 5.30 专业信息管理页面

第 6 章　WebService 和 Ajax 应用

本章要点：

- WebService 的创建、发布及调用
- Ajax 的概念及功能

技能目标：

- 会创建、发布及调用 WebService
- 会应用 Ajax 实现页面的无刷新

6.1　项目导入

【项目场景 1】

新建一个名为“ClothForSearch”的网站来模拟羽绒服总公司网站，再建一个名为“JXSCX”的网站，来模拟各分公司网站，通过 WebService 查询羽绒服库存量，如图 6.1 所示。

【项目场景 2】

实现登录功能，输入用户名和密码，提示验证结果，整个过程页面无刷新，如图 6.2 所示。

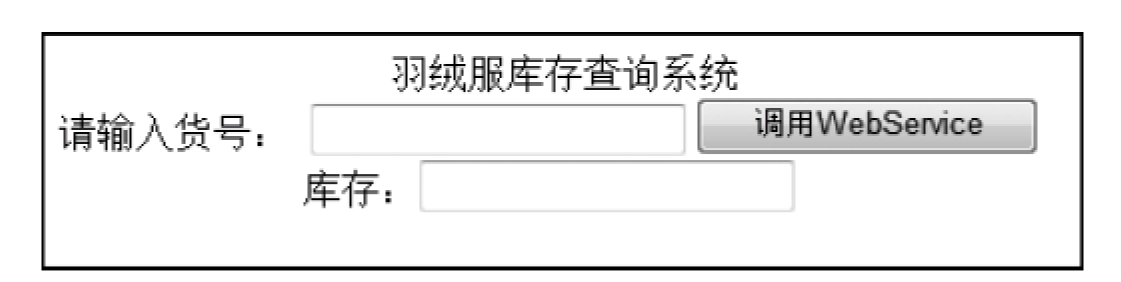


图 6.1　羽绒服库存查询系统



图 6.2　登录系统

【问题引导】

(1) WebService的概念是什么。
(2) 如何创建、发布及调用WebService。
(3) Ajax的概念及功能是什么。
(4) Ajax如何实现页面的无刷新。

6.2 技术与知识准备

6.2.1 WebService技术实现方法

6.2.1.1 WebService概述

如今软件开发的趋势是将软件视为一种基础设施与服务,这就是SaaS(Software as a Service,软件即服务)的理念。以此催生出了SOA(Service Oriented Architecture,面向服务的架构),以服务作为软件的基本构造块来"组装"软件,Web Service是当前最成熟的SOA技术之一,它可以将网站的功能封装成可复用软件服务,供其他应用程序调用。

Web Service也叫XML Web Service。WebService是一种可以接收从Internet或者Intranet上的其他系统中传递过来的请求,轻量级的独立的通讯技术。是通过SOAP在Web上提供的软件服务,使用WSDL文件进行说明,并通过UDDI注册。

XML(Extensible Markup Language,扩展型可标记语言),面向短期的临时数据处理、面向万维网络,是Soap的基础。

SOAP(Simple Object Access Protocol,简单对象存取协议),是XML Web Service的通信协议。当用户通过UDDI找到你的WSDL描述文档后,他可以通过SOAP调用你建立的Web服务中的一个或多个操作。SOAP是XML文档形式的调用方法的规范,它可以支持不同的底层接口,像HTTP(S)或者SMTP。

WSDL(Web Services Description Language)文件是一个XML文档,用于说明一组SOAP消息以及如何交换这些消息。大多数情况下由软件自动生成和使用。

UDDI(Universal Description, Discovery, and Integration)是一个主要针对Web服务供应商和使用者的新项目。在用户能够调用Web服务之前,必须确定这个服务内包含哪些商务方法,找到被调用的接口定义,还要在服务端来编制软件,UDDI是一种根据描述文档来引导系统查找相应服务的机制。UDDI利用SOAP消息机制(标准的XML/HTTP)来发布、编辑、浏览以及查找注册信息。它采用XML格式来封装各种不同类型的数据,并且发送到注册中心或者由注册中心返回需要的数据。

6.2.1.2 WebService技术实现

【示例6.1】 根据用户名查询用户密码,用WebService实现。

1. 创建 Web Service

【步骤 1】启动 Visual Studio2012，选择“文件”→“新建”→“网站”，建立名为 xcjweb 的网站。

【步骤 2】添加数据库 MyClassDB 和数据表 UserInfos，如图 6.3 所示。

OEM-20130723TNF....B - dbo.UserInfos

列名	数据类型	允许 Null 值
UserId	int	☐
UserName	nvarchar(50)	☑
UserPassword	nvarchar(50)	☑
UserType	nvarchar(50)	☑
		☐

图 6.3　UserInfos 表

【步骤 3】右击网站，依次选择“添加”→“Web 服务”，如图 6.4 所示，弹出如图 6.5 所示的对话框，输入项名称，单击“确定”按钮。添加成功后，会发现 xcjweb 网站中增加了 WebService.asmx 文件，同时在 App_Code 文件夹下面增加了 WebService.cs 文件。



图 6.4　添加 Web 服务

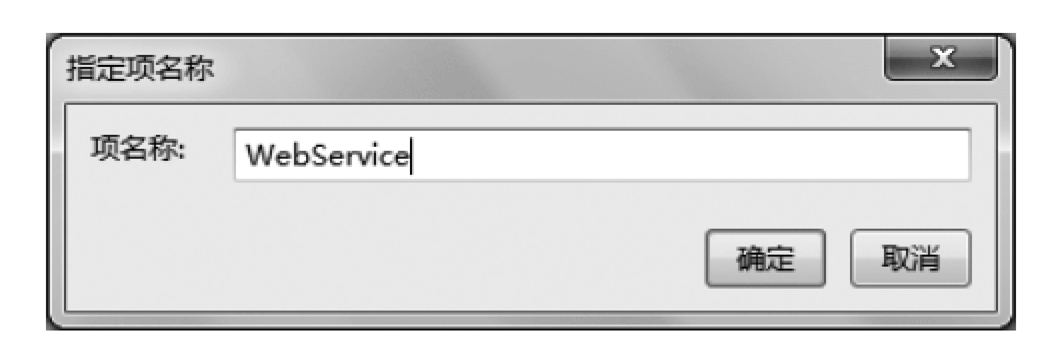


图 6.5　修改项名称

新建的 WebService.cs 有如下默认代码：

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
```

```
using System.Web.Services;

///<summary>
///WebService 的摘要说明
///</summary>
[WebService(Namespace="http://tempuri.org/")]
[WebServiceBinding(ConformsTo=WsiProfiles.BasicProfile1_1)]
//若允许使用 ASP.NET AJAX 从脚本中调用此 Web 服务,请取消注释以下行。
//[System.Web.Script.Services.ScriptService]
public class WebService: System.Web.Services.WebService {
    public WebService(){
        //如果使用设计的组件,请取消注释以下行
        //InitializeComponent();
    }
    [WebMethod]
    public string HelloWorld(){
        return "Hello World";
    }
    }
```

说明：如果要使方法成为 WebService 方法，必须在该方法前面加上[WebMethod]。

【步骤 4】编写代码。

配置 Web.config。

```
<connectionStrings>
    <add name =" DefaultConnection " providerName =" System. Data. SqlClient "
    connectionString="Data Source=OEM-20130723TNF;Initial Catalog=MyClassDB;User ID=
   sa;Password=123456" />
  </connectionStrings>
```

在 WebService.cs 编写代码如下：

```
[WebMethod]
    public string GetPwdByName(string name){
        string connString=ConfigurationManager.ConnectionStrings
        ["DefaultConnection"].ConnectionString;
        string sql=string.Format("select UserPassword from dbo.UserInfos where UserName
        ='{0}'", name);
        SqlConnection connection=new SqlConnection(connString);
        SqlCommand command=new SqlCommand(sql, connection);
        connection.Open();
        string pwd=(string)command.ExecuteScalar();
        connection.Close();
        return pwd;
    }
```

【步骤 5】右击解决方案资源管理器中的“xcjweb”，选择发布网站，打开如图 6.6 所示的对话框，选择好目标位置，单击“确定”按钮（在桌面上新建文件夹 xcj，发布的网站目标位置为该文件夹）。

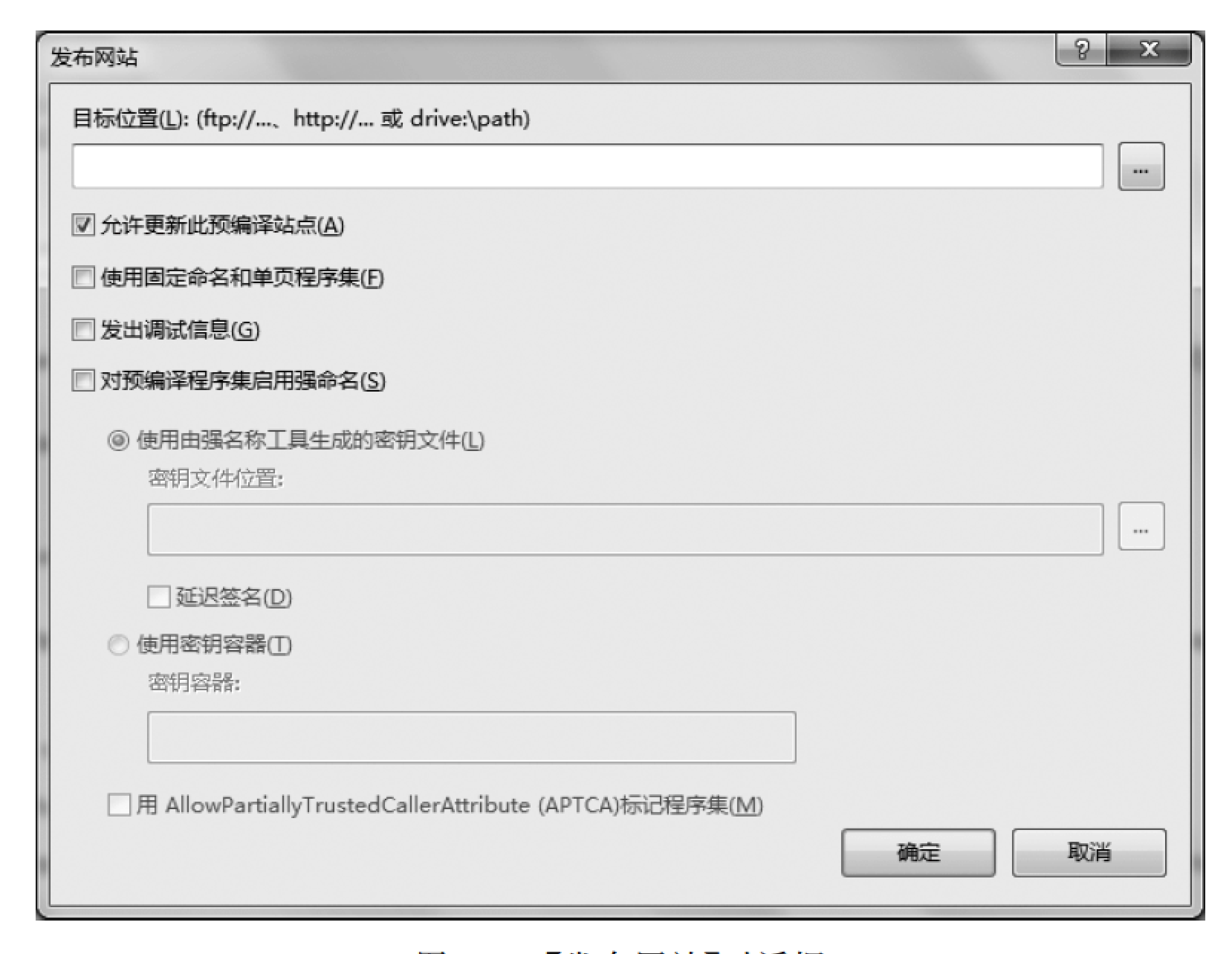


图 6.6　【发布网站】对话框

2. 发布 Web Service

【步骤 1】配置环境

（1）打开控制面板，单击“程序和功能”，弹出如图 6.7 所示的对话框，单击上方“打开或关闭 Windows 功能”，弹出图 6.8 所示的对话框，参照图 6.8 所示进行配置。



图 6.7　【程序和功能】对话框



图 6.8　【打开或关闭 Windows 功能】对话框

(2) 打开控制面板，单击“管理工具”，弹出如图 6.9 所示的对话框，双击“Internet 信息服务(IIS)管理器”，弹出如图 6.10 所示的对话框。

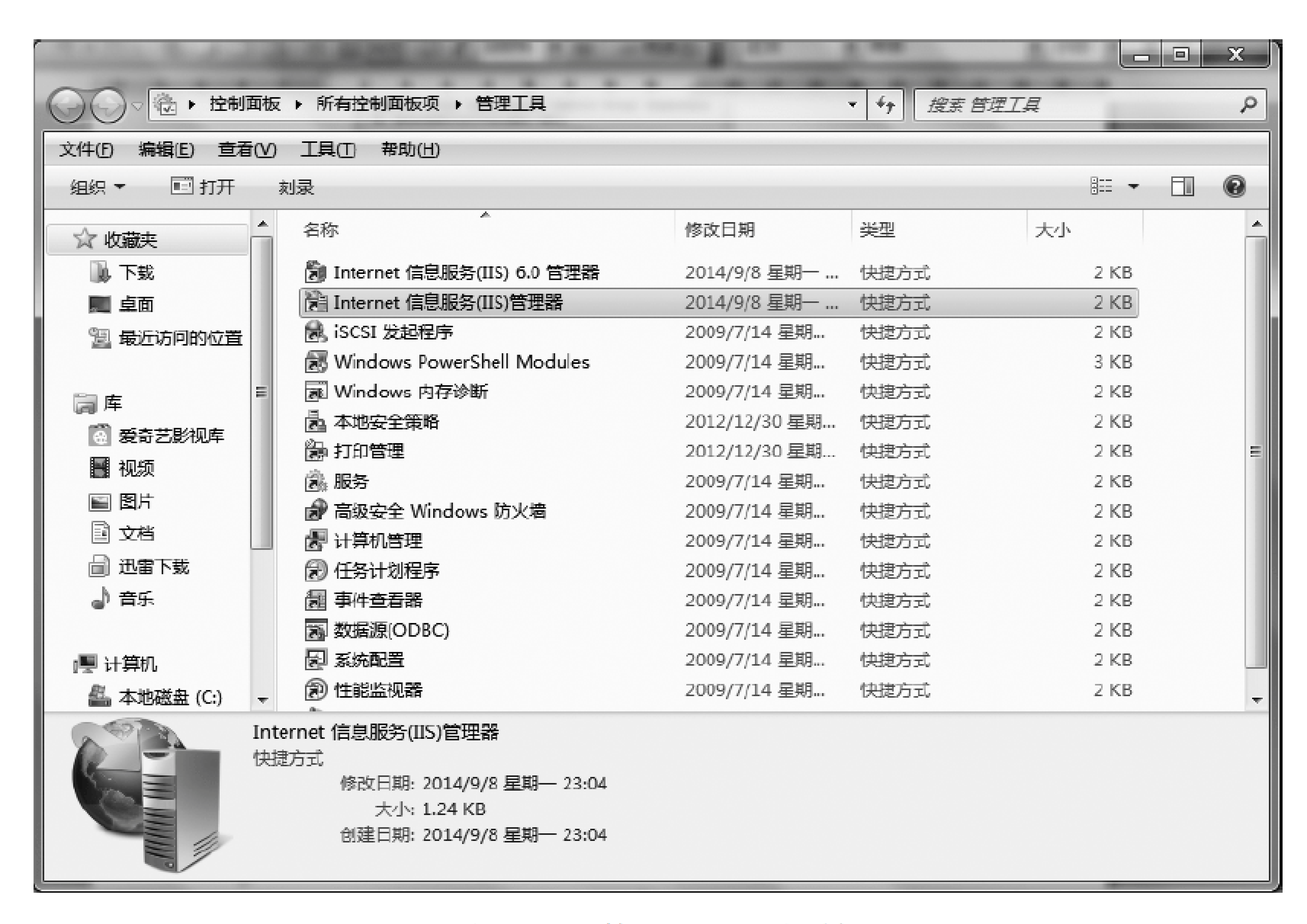


图 6.9　【管理工具】对话框

(3) 右击左侧“Default Web Site”，依次选择“管理网站”→“高级设置”，弹出图 6.11 所示的对话框。

(4) 修改应用程序池，单击后面的[...]，弹出图 6.12 所示的对话框，选择 Classic .NET AppPool。



图 6.10　【Internet 信息服务】对话框

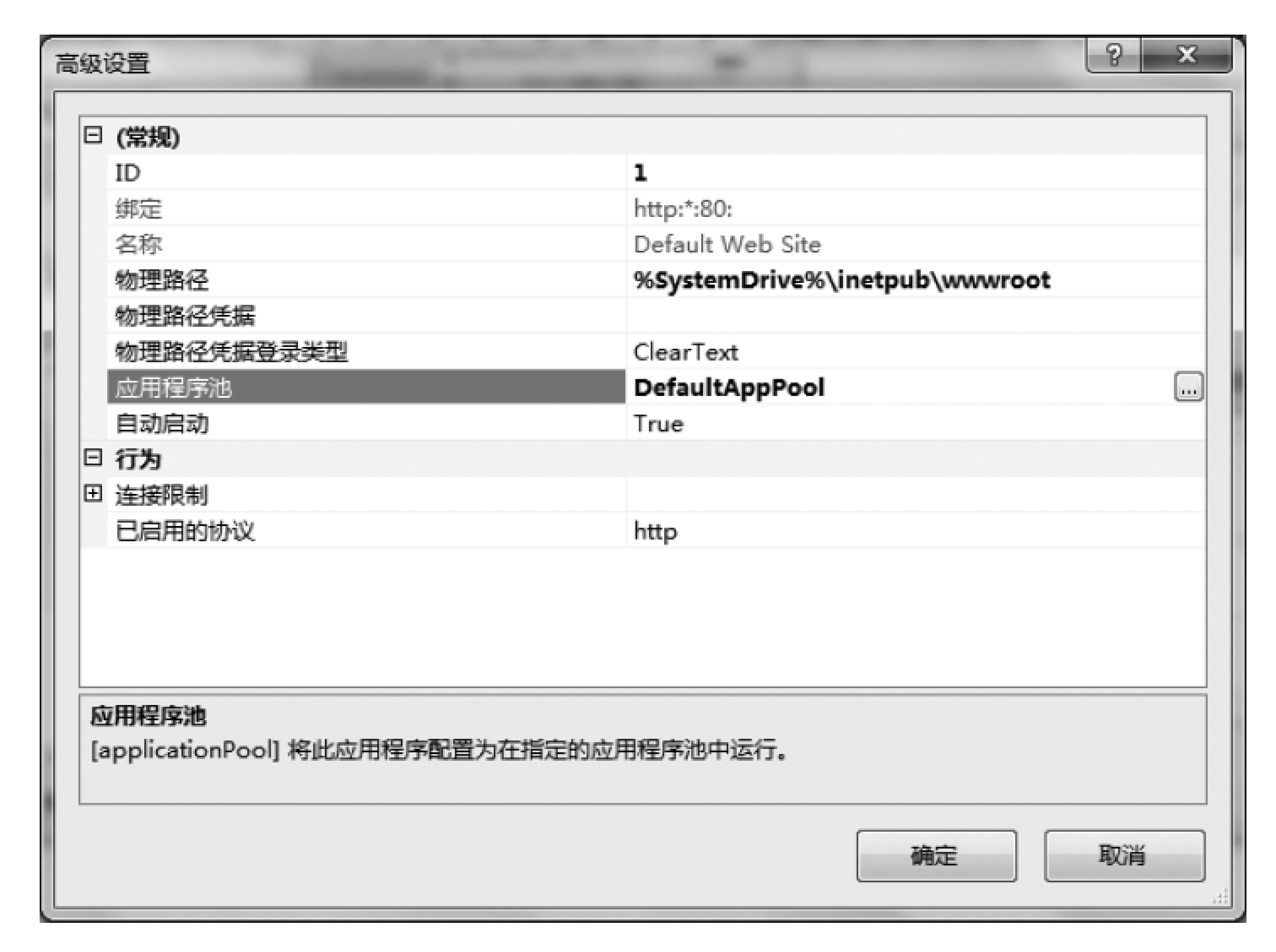


图 6.11　【高级设置】对话框

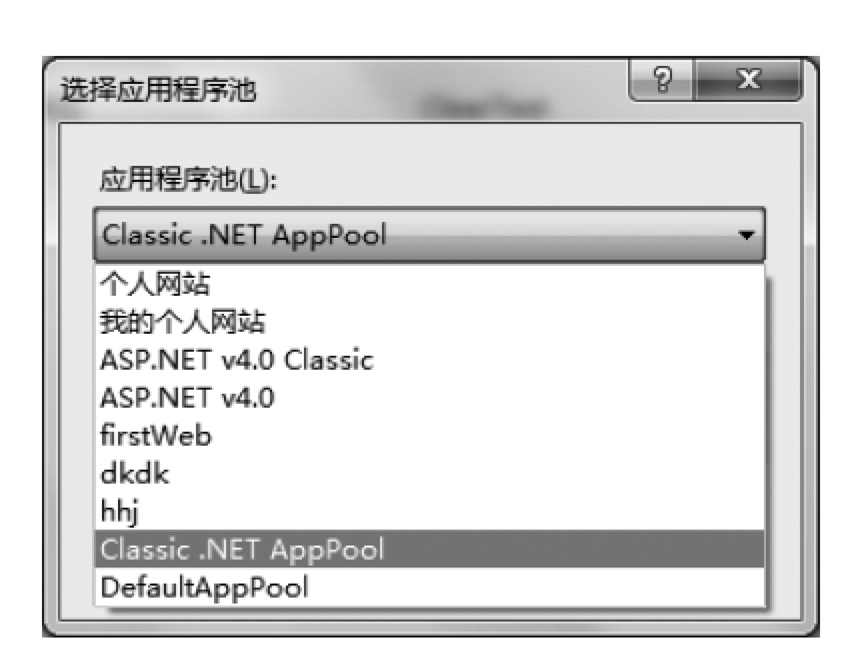


图 6.12　【选择应用程序池】对话框

（5）打开“Internet 信息服务(IIS)管理器”，双击左侧的“应用程序池”，右侧显示应用程序池列表，右击“Classic .NET AppPool”，选择“高级设置”，弹出图 6.13 所示的对话

框，在“进程模型”的“标识”中选择“NetworkService”。



图 6.13 【高级设置】对话框

（6）在“Internet 信息服务(IIS)管理器”界面中，右击“网站”，单击“添加网站”，弹出图 6.14 所示的对话框。



图 6.14 【添加网站】对话框

（7）对对话框进行如图 6.15 所示的配置。

图 6.15　配置【添加网站】对话框

（8）单击“Internet 信息服务(IIS)管理器”右侧的“编辑权限”，弹出如图 6.16 所示的对话框，单击“安全”选项卡，单击“编辑”按钮，再单击“添加”按钮，弹出如图 6.17 所示的对话框，在“输入对象名称来选择”下框中输入“Everyone”，单击“确定”按钮。

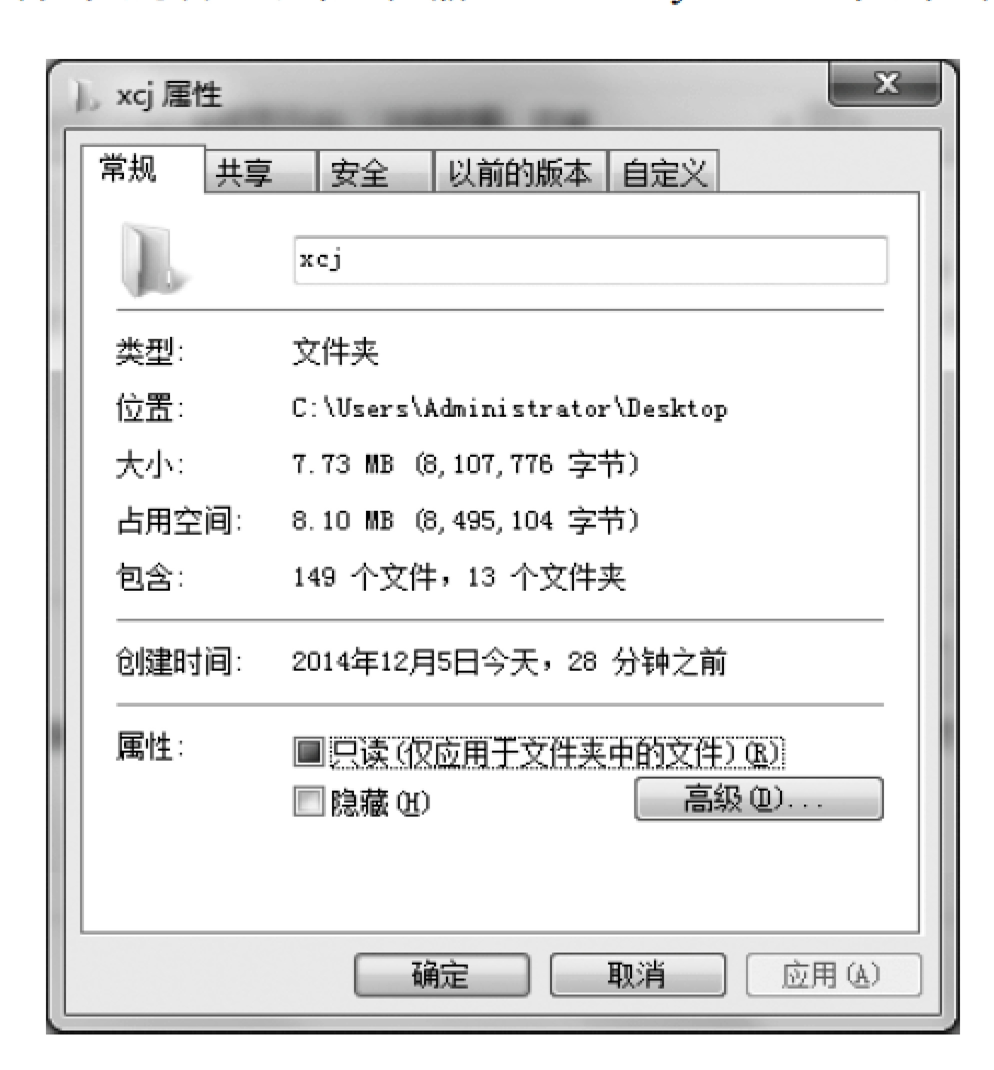


图 6.16　【xcj 属性】对话框



图 6.17 【选择用户或组】对话框

【步骤 2】测试 WebService。

(1) 单击“Internet 信息服务(IIS)管理器”右侧的“浏览 192.168.1.102:80(http)”,打开 IE 浏览器,在地址栏中添加 Web 服务的名称 WebService.asmx,弹出图 6.18 所示的网页。

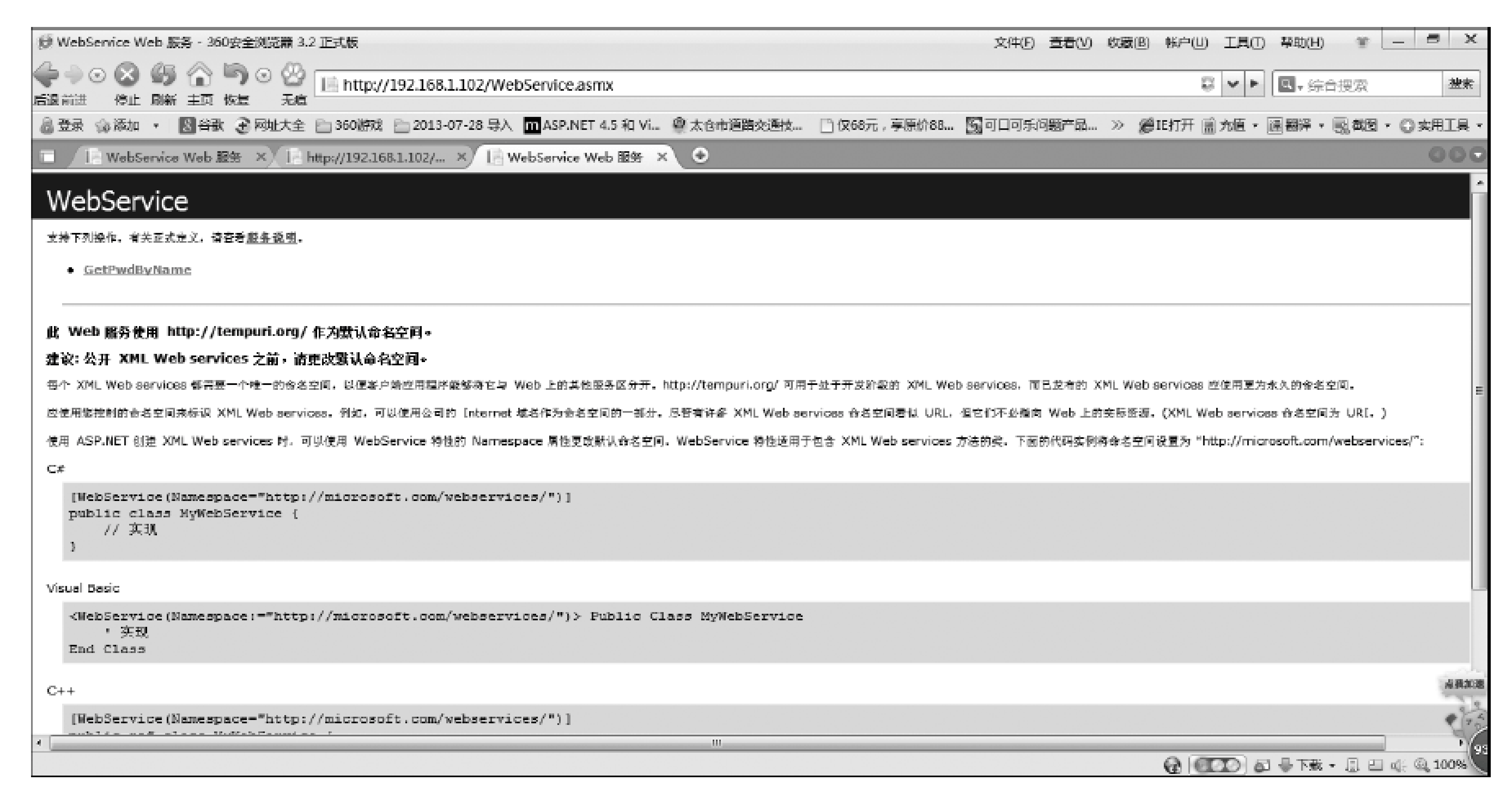


图 6.18 WebService 页面

(2) 单击“GetPwdByName”,弹出图 6.19 所示的页面,在编辑框中输入“cc”,单击“确定”,弹出图 6.20 所示的页面。

3. 调用 Web Service

【步骤 1】建立网站。

新建一个名为“symweb”的网站,并添加“Default2.aspx”的页面,在页面上添加两个文本控件和一个按钮控件,源视图代码如下:

```
<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
<meta http-equiv="Content-Type" content="text/html; charset=utf-8"/>
    <title></title>
```



图 6.19　输入参数页面



图 6.20　数据返回页面

```
</head>
<body>
    <form id="form1" runat="server">
    <div>
        用户名:<asp:TextBox ID="TextBox1" runat="server"></asp:TextBox>
        <asp:Button ID="Button1" runat="server" OnClick="Button1_Click" Text =
        "调用 WebService" />
        密码:<asp:TextBox ID="TextBox2" runat="server"></asp:TextBox>
    </div>
    </form>
</body>
</html>
```

【步骤 2】添加 Web 引用。

(1) 右击网站,选择"添加服务引用",弹出图 6.21 所示的对话框。

(2) 选择左下角的【高级】按钮,弹出"服务引用设置"的对话框,再单击左下角的【添加 Web 引用】,弹出"添加 Web 引用"对话框,如图 6.22 所示。

(3) 在 URL 中输入之前成功发布的 WebService 网站:"http://192.168.1.101/WebService.asmx";单击后面的转到按钮,能够看到 GetPwdByName 方法,修改 Web 引

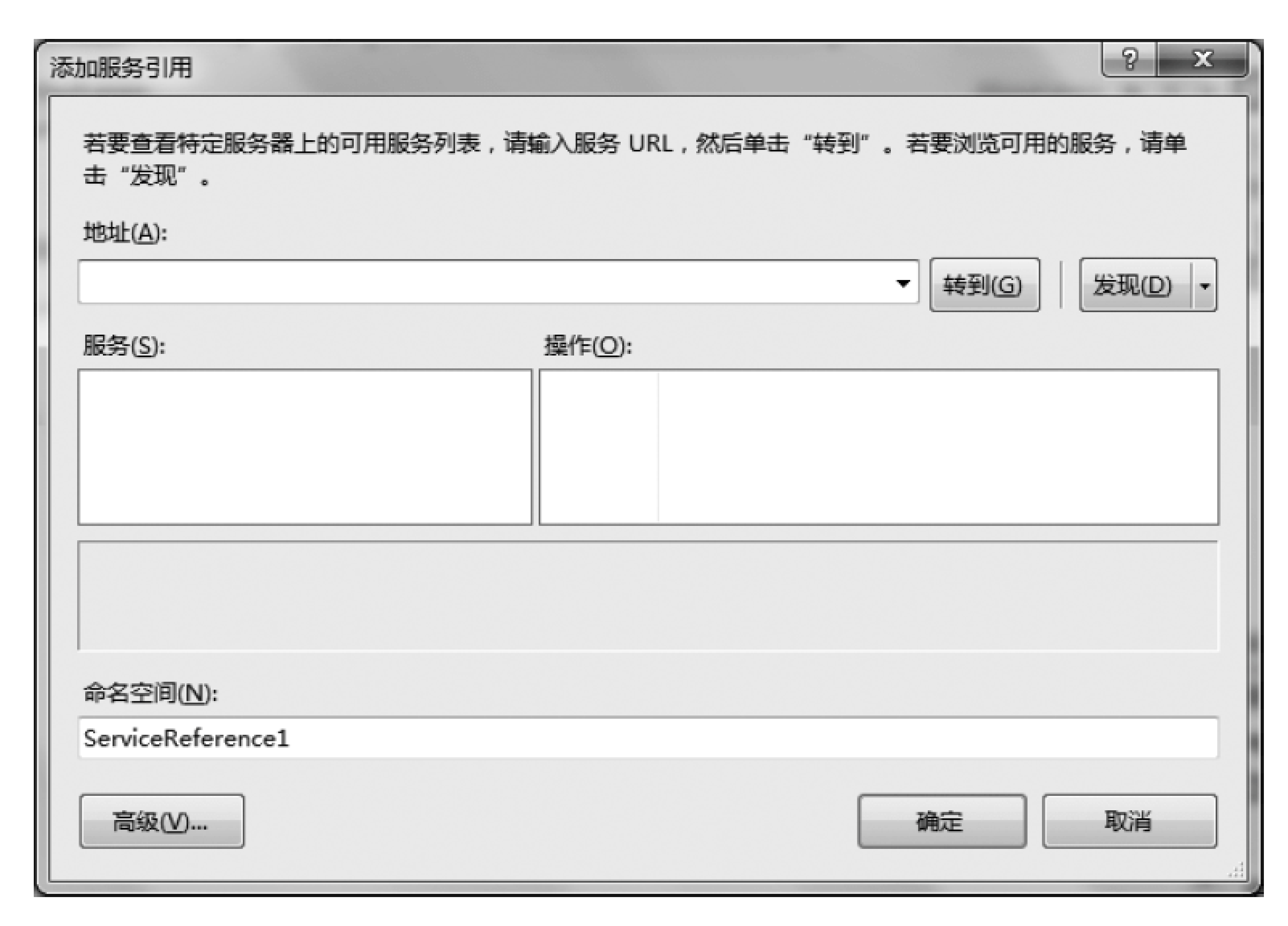


图 6.21　添加服务引用

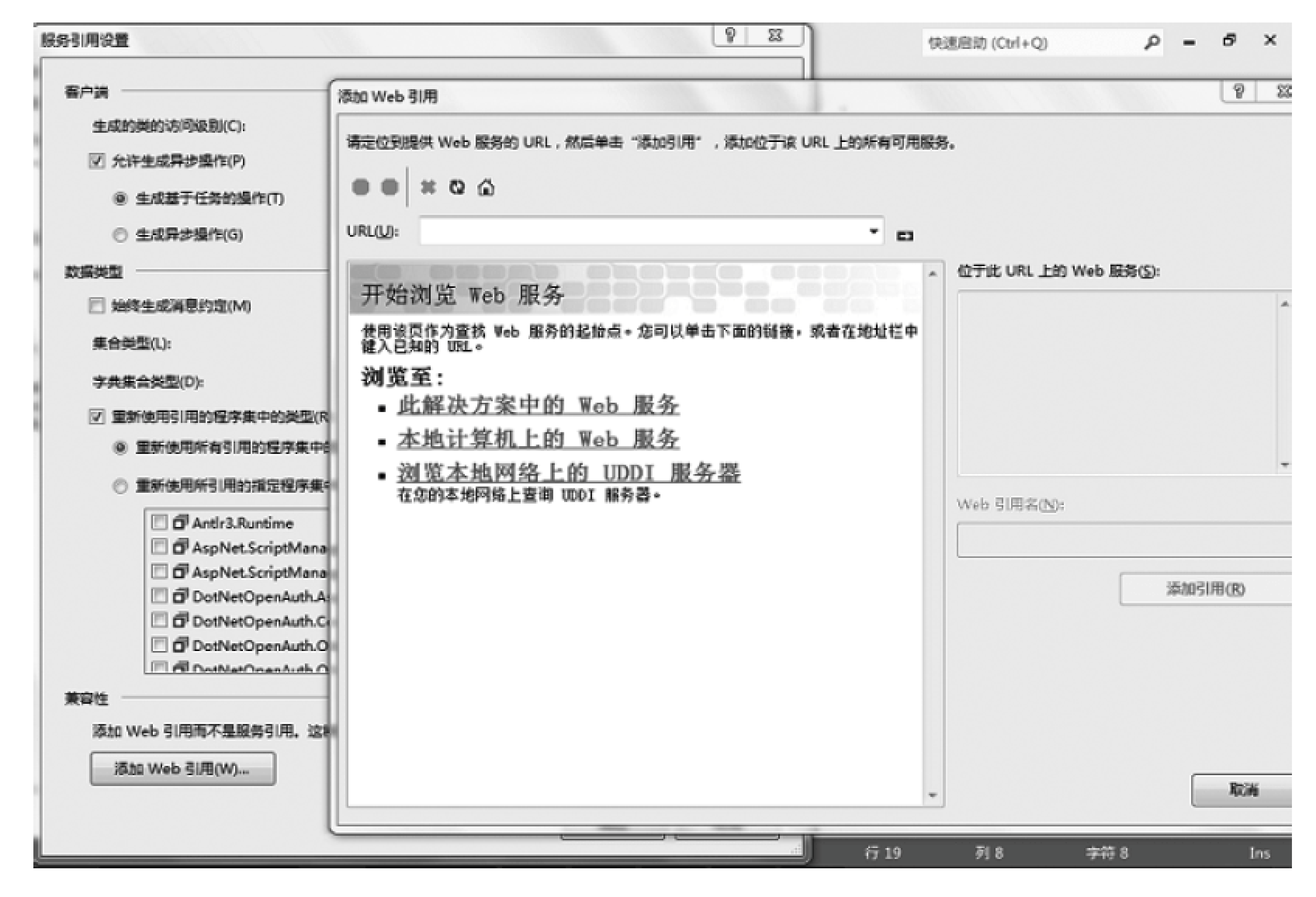


图 6.22　添加 Web 引用

用名，如图 6.23 所示，单击添加引用按钮，这时可以看到在网站中添加了相关文件夹。

【步骤 3】实现调用 WebService。

代码如下所示：

```
using System.Web;
using System.Web.UI;
using System.Web.UI.WebControls;
```



图 6.23 WebService 查找

```
using WebReference;                              //引用命名空间

public partial class Default2 : System.Web.UI.Page
{
    protected void Page_Load(object sender, EventArgs e)
    {
    }
    protected void Button1_Click(object sender, EventArgs e)
    {
        WebService service=new WebService();     //类创建实例
        TextBox2.Text=service.GetPwdByName(TextBox1.Text.Trim());
    }
}
```

6.2.2 Ajax 技术实现方法

Ajax 即“Asynchronous Javascript And XML”(异步 JavaScript 和 XML)，是指一种创建交互式网页应用的网页开发技术。Ajax 是“客户端＋服务器端”技术，通过在后台与服务器进行少量数据交换，Ajax 可以使网页实现异步更新。这意味着可以在不重新加载整个网页的情况下，对网页的某部分进行更新。

Javascript 和 XML 是 Ajax 技术中重要技术元素，但是 Ajax 技术远不止这两项内容，还包括 XMLHttpRequest 数据交换对象和 DOM 文档对象等技术内容。

【示例 6.2】 在大润发超市，用户根据超市卡卡号，查询余额。

1. 实现服务器端的程序

基本思路：通过 URL 接收一个超市卡卡号，然后调用相关代码判断，返回对应的余额。

Ajax 服务器端程序的实现技术平台没有限制，使用 java、.NET、PHP 等技术都没有任何问题。我们将使用 ASP.NET 程序作为服务器端。ASP.NET 服务器端可以是普通的页面，也可以是一般处理程序。在本例中，使用一般处理程序作为 Ajax 程序的服务器端。



图 6.24 搭建系统框架

【步骤 1】搭建系统架构，并添加各层之间的引用，如图 6.24 所示。

【步骤 2】新建数据库 TCD，添加数据表 Tcards，如图 6.25 所示。配置 Web.Config：“<add name="DefaultConnection" providerName="System.Data.SqlClient" connectionString="Data Source=OEM-20130723TNF; Initial Catalog=TCD; User ID=sa;Password=123456" />”。

【步骤 3】根据数据表添加相应类，如图 6.26 所示。

OEM-20130723TNF.TCD - dbo.Tcards

列名	数据类型	允许 Null 值
CNo	nvarchar(50)	☑
Mon	money	☑
		☐

图 6.25 数据表

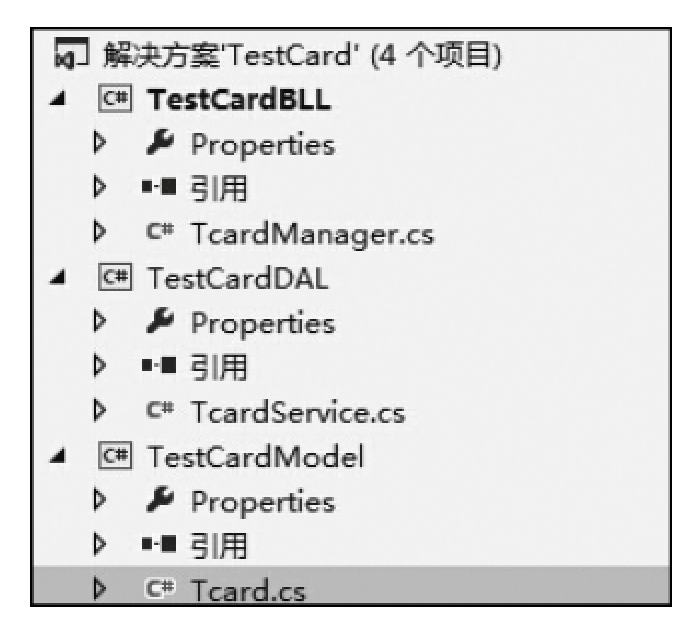


图 6.26 添加类库

【步骤 4】编写实体层代码。

```
public class Tcard
    {
        private string cNo;
        public string CNo
        {
            get { return cNo; }
            set { cNo=value; }
        }
        private decimal mon;
        public decimal Mon
        {
```

```
            get { return mon; }
            set { mon=value; }
        }
    }
```

【步骤5】编写数据访问层代码。

```
public decimal GetMonByNo(string cno)
    {
        string connString=ConfigurationManager.ConnectionStrings
        ["DefaultConnection"].ConnectionString;
        SqlConnection connection=new SqlConnection(connString);
        string sql=string.Format("select Mon from Tcards where CNo='{0}'",cno);
        SqlCommand command=new SqlCommand(sql, connection);
        connection.Open();
        decimal money=(decimal)command.ExecuteScalar();
        connection.Close();
        return money;
    }
```

【步骤6】编写业务逻辑层代码。

```
public decimal GetMonByNo(string cno)
    {
        return new TcardService().GetMonByNo(cno);
    }
```

【步骤7】右击TestCardWeb,依次选择“添加”→“添加新项”,弹出“添加新项”对话框,选择“C#”和“一般处理程序”,在名称框中输入“TCardHandler.ashx”,如图6.27所示,单击“添加”按钮。



图6.27 添加一般处理程序

【步骤 8】编写一般处理程序代码。

```
<%@ WebHandler Language="C#" Class="TCardHandler" %>
using System;
using System.Web;
using TestCardBLL;
public class TCardHandler : IHttpHandler {
        public void ProcessRequest(HttpContext context){
        context.Response.ContentType="text/plain";
        string id=context.Request.QueryString["CID"];      //接收客户端传递过来的卡号
        decimal mon=new TcardManager().GetMonByNo(id);
        context.Response.Write("您的余额为"+mon);
    }
     public bool IsReusable {
        get {
            return false;
        }
    }
}
```

2. 实现 Ajax 客户端

新建页面 CardCX.aspx,编写代码如下:

```
<script type="text/javascript">
        var xhr;
        function createXMLHttpRequest(){
            if(window.ActiveXObject){              //如果是 IE 浏览器
                return new ActiveXObject("Microsoft.XMLHTTP");
            }
            else if(window.XMLHttpRequest){        //非 IE 浏览器
                return new XMLHttpRequest();
            }
        }
        function search(){
            var cno=document.getElementById("txtCNO").value;
            if(cno !=""){
                //请求字符串
                var url="TCardHandler.ashx?CID="+cno.toString();
                //1. 创建 XMLHttpRequest 组件
                xhr=createXMLHttpRequest();
                //2. 设置回调函数
                xhr.onreadystatechange=readyDo;
                //3. 初始化 XMLHttpRequest 组件
                xhr.open("GET", url, true);
                //4. 发送请求
```

```
                xhr.send(null);
            }
        }
        function readyDo(){
            if(xhr.readyState==4
             && xhr.status==200){
                alert(xhr.responseText);
            }
        }
</script>
<table>
        <tr><td colspan="2">大润发查询系统</td></tr>
        <tr><td>请输入大润发卡号</td><td><input id="txtCNO" type="text" />
        </td></tr>
        <tr><td colspan="2"><input id="Button1" type="button" value="查询" onclick="
        search()" /></td></tr>
    </table>
```

运行结果如图 6.28 所示。

说明：

1. 在 JavaScript 中，XMLHttpRequest 可作为一个对象来使用，由于不同浏览器创建该对象的方式不一致，所以可以使用一个函数（本例中使用 createXMLHttpRequest）专门负责创建该对象。

2. 使用 XMLHttpRequest 访问服务器端基本分为四个步骤：

（1）创建 XMLHttpRequest 对象。

（2）设置回调函数。

回调函数用来对服务器端返回结果进行相应处理。



图 6.28　调用 Ajax 运行结果

（3）初始化 XMLHttpRequest 组件。

有三个参数：第一个是设置所用方法 GET 或 POST，第二个是目标资源 URL 的字符串，第三个是指示请求是否异步。

（4）发送请求。

使用 send 方法向服务器发送请求。

3. 在获取服务器端返回值之前，先要对 XMLHttpRequest 对象的状态进行判断，要判断 XMLHttpRequest 是否已经接收到服务器端返回数据，readyState 值为 4 时表示数据接收完毕，status 为 200 表示正确返回。

4. 在单击查询按钮时，激发 click 事件，调用 search()函数。

6.3 项 目 训 练

6.3.1 项目训练 1

通过对以上内容的学习，了解了 WebService 的概念及功能，同时了解了创建、发布及调用 WebService 的方法，现在我们回到项目导入的任务中来。

1. 创建 WebService

【步骤 1】新建 ClothForSearch 的网站来模拟总公司网站，并添加好数据库，如图 6.29 所示。

【步骤 2】添加 Web 服务，如图 6.30 所示。



图 6.29 ClothAmount 数据表

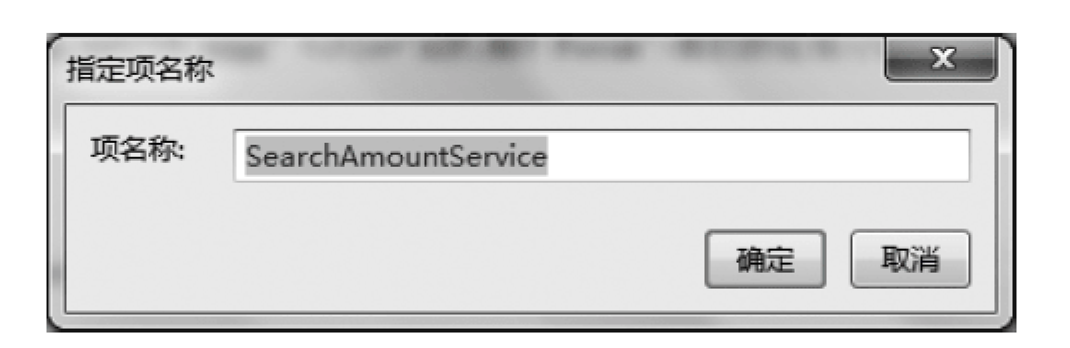


图 6.30 添加 Web 服务

【步骤 3】编写代码。

```
[WebMethod]
    public int GetAmountById(string id){
        string connString=ConfigurationManager.ConnectionStrings
        ["DefaultConnection"].ConnectionString;
        string sql=string.Format("select Amount from dbo.ClothAmount where ID= '{0}'",
        id);
        SqlConnection connection=new SqlConnection(connString);
        SqlCommand command=new SqlCommand(sql, connection);
        connection.Open();
        int amount=(int)command.ExecuteScalar();
        connection.Close();
        return amount;
    }
```

【步骤 4】发布网站到桌面的 ClothSearch 文件夹(已提前建好该文件夹)。

2. 发布 WebService

【步骤 1】前面 IIS 已配置，这里略。打开 Internet 信息服务(IIS)管理器，右击网站，选择添加网站，弹出“添加网站”对话框，进行如图 6.31 的设置(这里，可以先把上文建的网站停止，方法：右击个人网站，依次选择“管理网站”→“停止”即可)。

【步骤 2】单击“Internet 信息服务(IIS)管理器”右侧的“编辑权限”，添加“Everyone”用户名。



图 6.31　添加网站

【步骤 3】单击“Internet 信息服务(IIS)管理器”右侧的“浏览 192.168.1.102：80 (http)”，打开 IE 浏览器，在地址栏中添加 Web 服务的名称 SearchAmountService.asmx，弹出图 6.32 所示的网页。



图 6.32　WebService 页面

【步骤 4】在编辑框中输入 id 号，单击调用按钮，弹出图 6.33 所示的页面。

3. 调用 WebService

【步骤 1】新建一个名为“JXSCX”的网站，并添加“CXByWS.aspx”的页面，在页面上

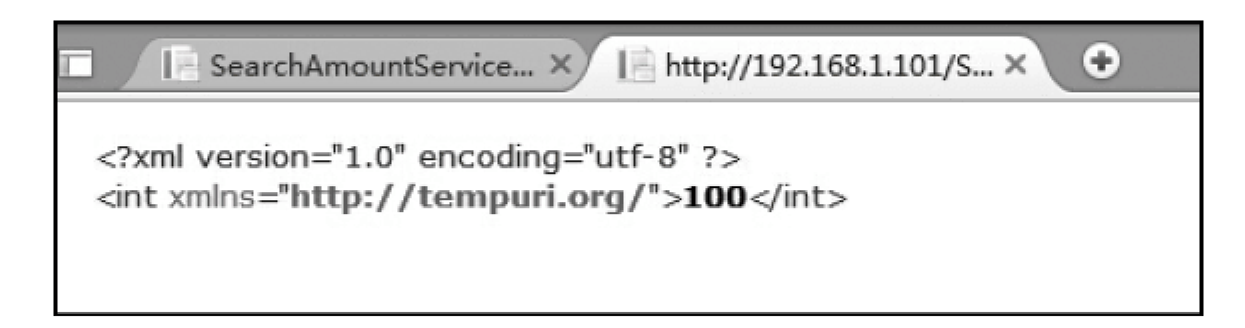


图 6.33 数据返回页面

添加两个文本控件和一个按钮控件,代码如下:

```
<%@ Page Language="C#" AutoEventWireup="true" CodeFile="CXByWS.aspx.cs" Inherits="
CXByWS" %>
<!DOCTYPE html>
<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
<meta http-equiv="Content-Type" content="text/html; charset=utf-8"/>
    <title></title>
</head>
<body>
    <form id="form1" runat="server">
    <div>
           羽绒服库存查询系统<br />
       请输入货号:<asp:TextBox ID="TextBox1" runat="server"></asp:TextBox>
        <asp:Button ID="Button1" runat="server" OnClick="Button1_Click" Text="调用
        WebService" />
       库存:<asp:TextBox ID="TextBox2" runat="server"></asp:TextBox>
    </div>
    </form>
</body>
</html>
```

【步骤 2】添加 Web 引用。

(1) 右击网站,选择"添加服务引用",弹出"添加服务引用"对话框,单击左下角"高级",弹出"服务引用设置"对话框,单击左下角"添加 Web 引用"按钮,弹出"添加 Web 引用"对话框,如图 6.34 所示。

(2) 在 URL 中输入之前成功发布的 WebService 网站:"http://192.168.1.102/SearchAmountService.asmx";单击后面的转到按钮,能够看到方法 GetAmountById,修改 Web 引用名,单击"添加引用"按钮,如图 6.35 所示。

这时可以看到在网站中添加了相关文件夹。

(3) 实现调用 WebService。

代码如下所示:

```
using System;
using System.Collections.Generic;
using System.Linq;
```



图 6.34 WebService 查找

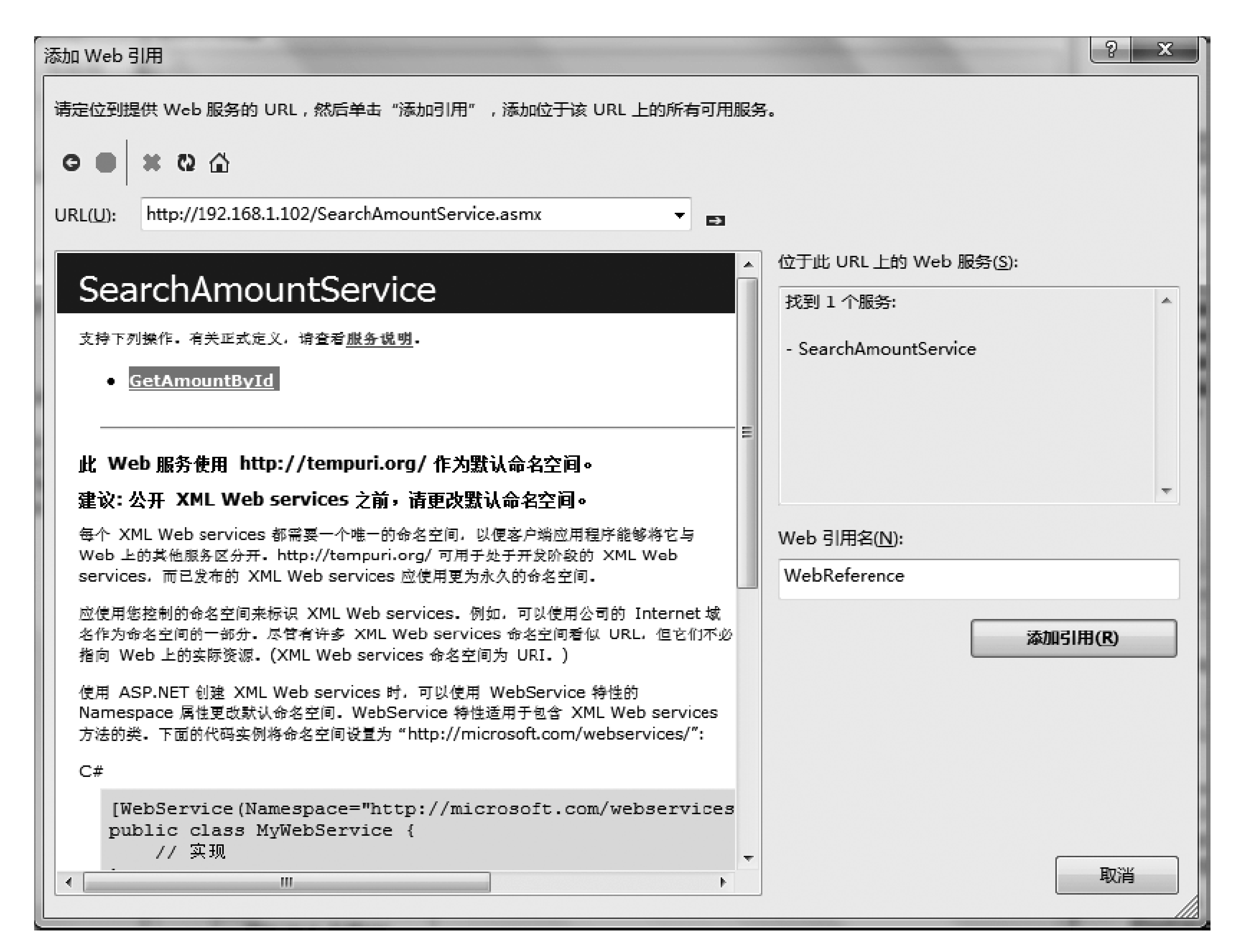


图 6.35 添加 Web 引用

```
using System.Web;
using System.Web.UI;
using System.Web.UI.WebControls;
using WebReference;                                    //引用命名空间
public partial class CXByWS : System.Web.UI.Page
```

```
{
    protected void Page_Load(object sender, EventArgs e)
    {
    }
    protected void Button1_Click(object sender, EventArgs e)
    {
        SearchAmountService saservice=new SearchAmountService();
        TextBox2.Text=saservice.GetAmountById(TextBox1.Text.Trim()).ToString();
    }
}
```

6.3.2　项目训练 2

通过对以上内容的学习，了解了 Ajax 的概念及原理，同时了解了使用 Ajax 实现页面无刷新的步骤，现在回到项目导入的任务中来。

1. 实现服务器端的程序

基本思路：通过 URL 接收用户名和密码，然后调用相关代码进行判断，返回对应的结果。

【步骤 1】搭建系统架构，并添加各层之间的引用，如图 6.36 所示。

【步骤 2】新建数据库 CLoginDB，添加数据表 tbCustomerLogins，如图 6.37 所示。配置 Web.Config："<add name="DefaultConnection" providerName="System.Data.SqlClient" connectionString="Data Source=OEM-20130723TNF;Initial Catalog=CLoginDB;User ID=sa;Password=123456" />"。

【步骤 3】根据数据表添加相应类，如图 6.38 所示。

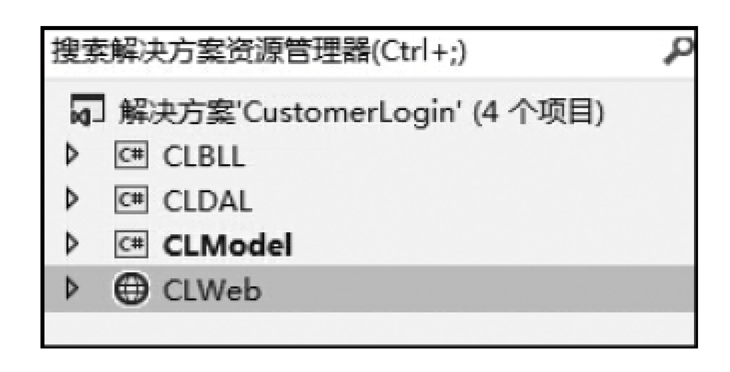


图 6.36　搭建系统框架

OEM-20130723TNF.....tbCustomerLogins

列名	数据类型	允许 Null 值
UserId	nvarchar(50)	☑
UserPwd	nvarchar(50)	☑

图 6.37　tbCustomerLogin 数据表



图 6.38　添加类

【步骤 4】编写实体层代码。

```
public class tbCustomerLogin
    {
        private string userId;
```

```
            public string UserId
            {
                get { return userId; }
                set { userId=value; }
            }
            private string userPwd;
            public string UserPwd
            {
                get { return userPwd; }
                set { userPwd=value; }
            }
        }
```

【步骤 5】编写数据访问层代码。

```
public bool GetByCout(string id, string pwd)
        {
                string connString=ConfigurationManager.ConnectionStrings
                ["DefaultConnection"].ConnectionString;
            SqlConnection connection=new SqlConnection(connString);
             connection .Open();
             string sql=string.Format("select count(*)from dbo.
             tbCustomerLogins where UserId='{0}' and UserPwd='{1}'",id,pwd);
             SqlCommand command=new SqlCommand(sql,connection);
             int n=(int)command.ExecuteScalar();
             connection.Close();
             return n>0;
        }
```

【步骤 6】编写业务逻辑层代码。

```
public bool GetByCout(string id, string pwd)
        {
            return new tbCustomerLoginService().GetByCout(id, pwd);
        }
```

【步骤 7】右击 CLWeb,依次选择"添加"→"添加新项",弹出"添加新项"对话框,选择"C#"和"一般处理程序",在名称框中输入"TLoginHandler.ashx",如图 6.39 所示。单击"添加"按钮。

【步骤 8】编写一般处理程序代码。

```
<%@ WebHandler Language="C#" Class="TLoginHandler" %>
using System;
using System.Web;
using CLBLL;
using CLModel;
```



图 6.39　添加一般处理程序

```
public class TLoginHandler : IHttpHandler {
        public void ProcessRequest(HttpContext context){
        context.Response.ContentType="text/plain";
        string id=context.Request.QueryString["id"];
        string pwd=context.Request.QueryString["pwd"];
        if(new tbCustomerLoginManager().GetByCout(id, pwd))
        context.Response.Write("用户名密码正确,登录成功");
        else
            context.Response.Write("用户名密码错误,登录失败");
    }
    public bool IsReusable {
        get {
            return false;
        }
    }
}
```

2. 实现 Ajax 客户端

新建页面 LoginCX.aspx,添加如下代码:

```
<%@ Page Language="C#" AutoEventWireup="true" CodeFile="LoginCX.aspx.cs" Inherits="
LoginCX" %>
<!DOCTYPE html>
<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
<meta http-equiv="Content-Type" content="text/html; charset=utf-8"/>
    <title></title>
    <script type="text/javascript">
```

```
        var xhr;
        function createXMLHttpRequest(){
            if(window.ActiveXObject){              //如果是 IE 浏览器
                return new ActiveXObject("Microsoft.XMLHTTP");
            }
            else if(window.XMLHttpRequest){        //非 IE 浏览器
                return new XMLHttpRequest();
            }
        }
        function search(){
            var id=document.getElementById("txtId").value;
            var pwd=document.getElementById("txtPwd").value;
            if(id !="" & pwd!=""){
                //请求字符串
                var url="TLoginHandler.ashx?id="+id.toString()+"&pwd="+pwd.toString
                ();
                //1.创建 XMLHttpRequest 组件
                xhr=createXMLHttpRequest();
                //2.设置回调函数
                xhr.onreadystatechange=readyDo;
                //3.初始化 XMLHttpRequest 组件
                xhr.open("GET", url, true);
                //4.发送请求
                xhr.send(null);
            }
        }
        function readyDo(){
            if(xhr.readyState==4
             && xhr.status==200){
                alert(xhr.responseText);
            }
        }
</script>
</head>
<body>
    <form id="form1" runat="server">
    <div style="text-align:center">
    <table>
        <tr><td colspan="2">登录系统</td></tr>
        <tr><td>用户名</td><td><input id="txtId" type="text" /></td></tr>
         <tr><td>密码</td><td><input id="txtPwd"  type="password" /></td></tr>
        <tr><td colspan="2"><input id="Button1" type="button" value="登录" onclick="
        search()" /></td></tr>
    </table>
    </div>
    </form>
```

```
</body>
</html>
```

6.4 平行项目训练

1. 训练内容

创建 WebService 实现用户名的模糊查询功能，另建一个网站，能够根据用户名查找员工信息。

2. 训练目的

（1）进一步训练和巩固学生对 WebService 的原理的理解；

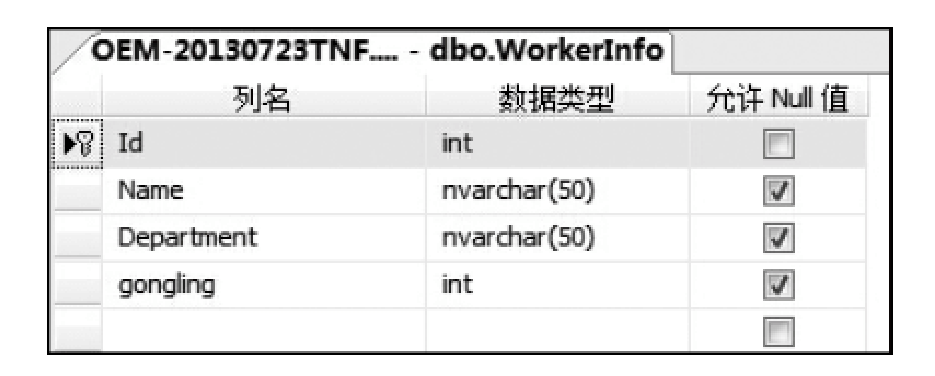
（2）使学生对 WebService 的创建、发布及调用有一个比较深刻的印象和认识。

3. 训练过程

【步骤 1】创建 WebService。

（1）新建 WorkerForSearch 的网站来模拟总公司网站，并添加数据库，如图 6.40 所示。

（2）添加 Web 服务，如图 6.41 所示。



列名	数据类型	允许 Null 值
Id	int	☐
Name	nvarchar(50)	☑
Department	nvarchar(50)	☑
gongling	int	☑
		☐

图 6.40 WorkerInfo 数据表

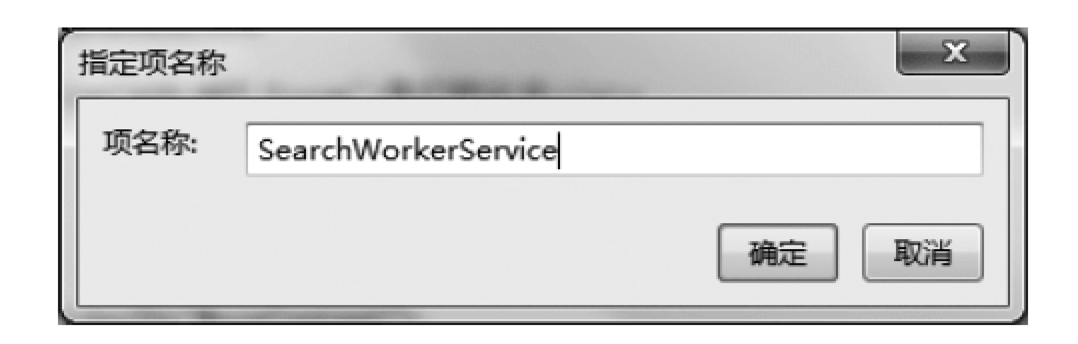


图 6.41 添加 Web 服务

（3）编写代码。

```
[WebMethod]
    public DataSet GetWorkerByName(string name){
        string connString=ConfigurationManager.ConnectionStrings
["DefaultConnection"].ConnectionString;
        string sql=string.Format("select * from dbo.WorkerInfo where Name like'%{0}%'",
        name);
        SqlConnection connection=new SqlConnection(connString);
        SqlDataAdapter da=new SqlDataAdapter(sql, connection);
        DataSet ds=new DataSet();
        da.Fill(ds);
        return ds;     }
```

（4）发布网站到桌面的 WorkerSearch 文件夹(已提前建好)。

【步骤 2】发布 WebService。

（1）前面 IIS 已配置，这里略。打开 Internet 信息服务(IIS)管理器，右击网站，选择

添加网站，弹出“添加网站”对话框，进行设置，如图 6.42 所示(这里，可以先把上文建的网站停止，方法：右击网站，依次选择“管理网站”→“停止”即可)。



图 6.42 添加网站

(2) 单击“Internet 信息服务(IIS)管理器”右侧的“编辑权限”，添加“Everyone”用户名。

(3) 单击“Internet 信息服务(IIS)管理器”右侧的“浏览 192.168.1.101：80(http)”，打开 IE 浏览器，在地址栏中添加 Web 服务的名称 SearchWorkerService.asmx，弹出图 6.43 所示的网页。

(4) 在编辑框中输入 name 号，单击调用按钮，弹出图 6.44 所示的页面。

【步骤 3】调用 WebService。

(1) 新建一个名为“WorkerCX”的网站，并添加“WorkerInfoCX.aspx”的页面，在页面上添加文本控件和按钮控件，代码如下：

```
<%@ Page Language="C#" AutoEventWireup="true" CodeFile="WorkerInfoCX.aspx.cs" Inherits
="WorkerInfoCX" %>
<!DOCTYPE html>
<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
<meta http-equiv="Content-Type" content="text/html; charset=utf-8"/>
    <title></title>
</head>
```

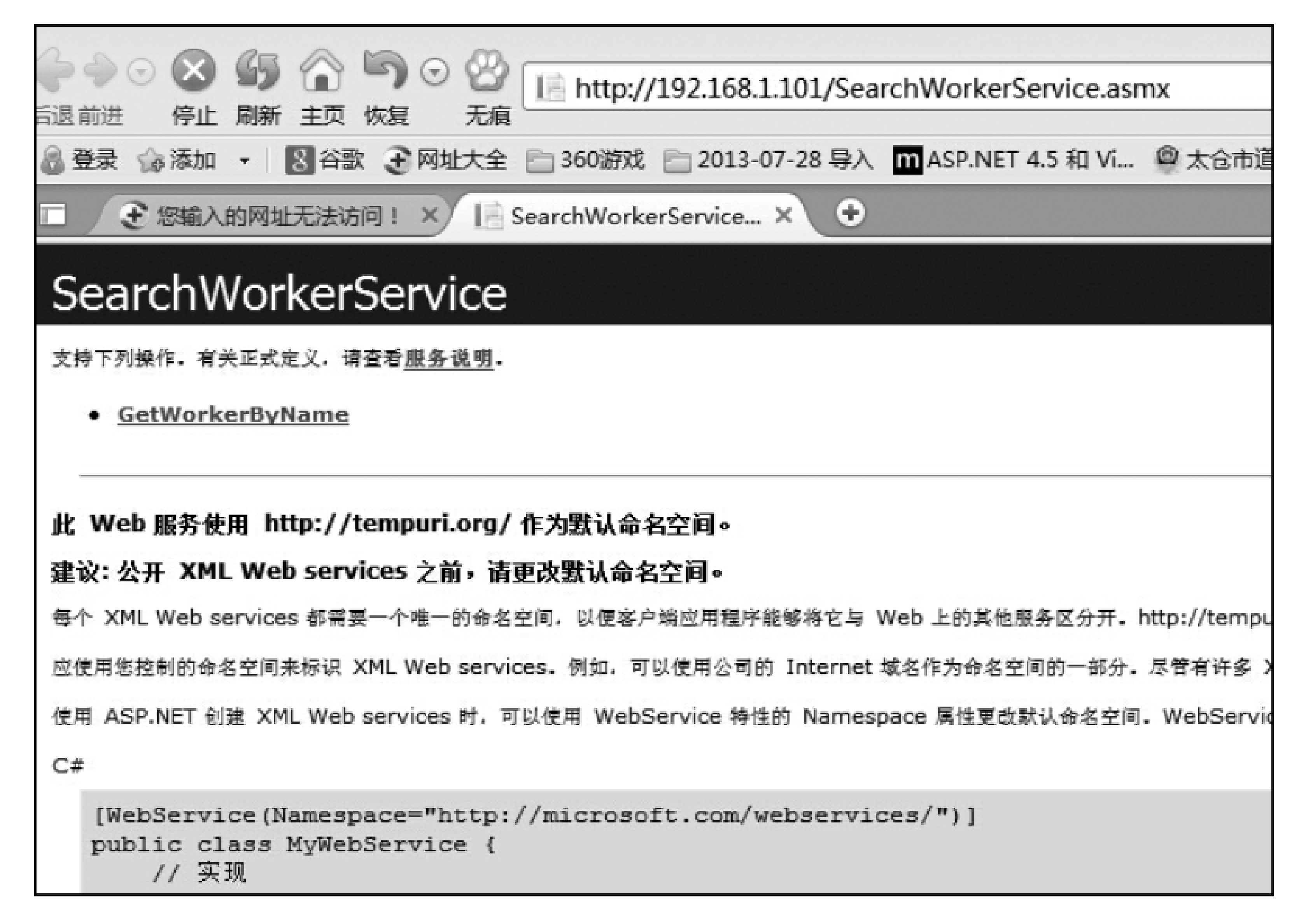


图 6.43　WebService 页面



图 6.44　数据返回页面

```
<body>
    <form id="form1" runat="server">
    <div>
    员工信息查询系统<br />
        请输入员工姓名(支持模糊查询)<asp:TextBox ID="txtName" runat="server"> </asp:
        TextBox><asp:Button ID="Button1" runat="server" Text="查询" /><br />
        <asp:GridView ID="GridView1" runat="server" AutoGenerateColumns= "False">
```

```
            <Columns>
                <asp:BoundField DataField="Name" HeaderText="姓名" />
                <asp:BoundField DataField="Department" HeaderText="部门" />
                <asp:BoundField DataField="gongling" HeaderText="工龄" />
            </Columns>
        </asp:GridView>
    </div>
    </form>
</body>
</html>
```

(2) 添加 Web 引用。

(a) 右击网站,选择"添加服务引用",弹出"添加服务引用"对话框,单击左下角"高级",弹出"服务引用设置"对话框,单击左下角"添加 Web 引用"按钮,弹出"添加 Web 引用"对话框,如图 6.45 所示。

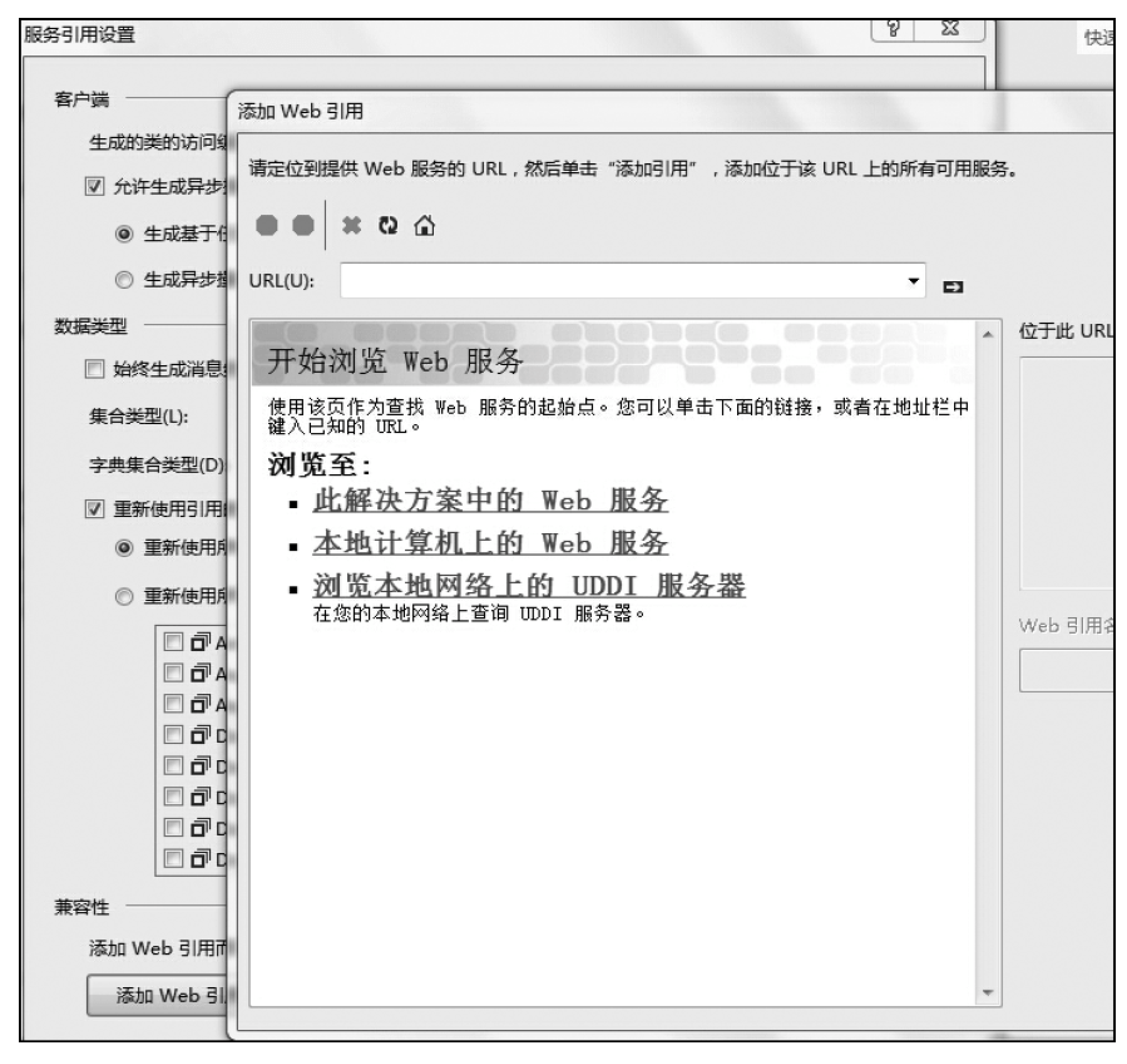


图 6.45 WebService 查找

(b) 在 URL 中输入之前成功发布的 WebService 网站:"http://192.168.1.102/SearchWorkerService.asmx";单击后面的"转到"按钮,能够看到 GetWorkerByName 方

法，修改 Web 引用名，单击“添加引用”按钮，这时可以看到在网站中添加了相关文件夹。如图 6.46 所示。

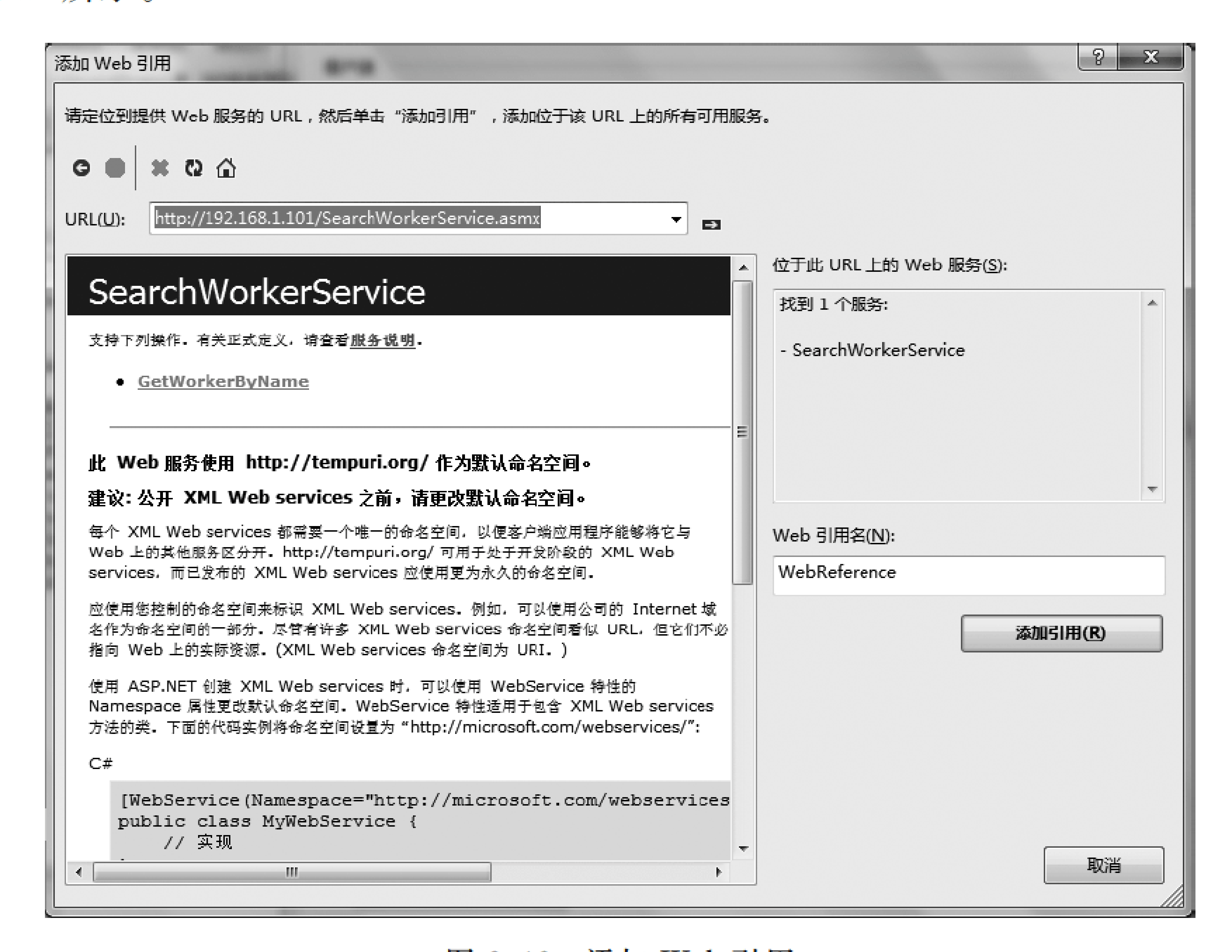


图 6.46 添加 Web 引用

（3）实现调用 WebService。

代码如下所示：

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.UI;
using System.Web.UI.WebControls;
using WebReference;

public partial class WorkerInfoCX : System.Web.UI.Page
{
    protected void Page_Load(object sender, EventArgs e)
    {
    }
    protected void Button1_Click(object sender, EventArgs e)
    {
        SearchWorkerService sws=new SearchWorkerService();
        GridView1.DataSource=sws.GetWorkerByName(txtName.Text.Trim());
        GridView1.DataBind();
    }
```

```
}
```

运行结果如图6.47所示。

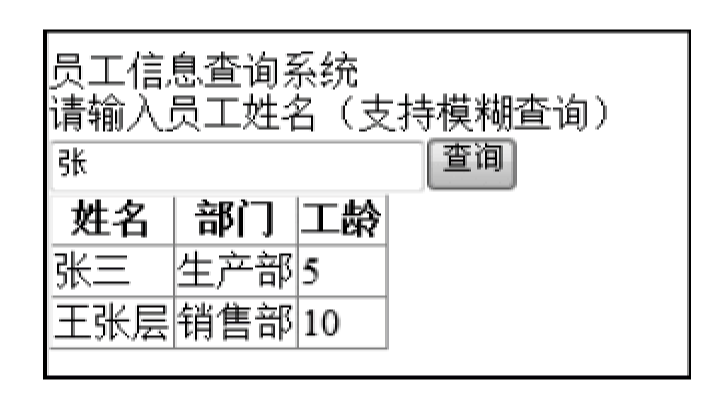


图6.47　查询页面

6.5　总　　结

本章通过简单项目案例，介绍了WebService的原理及其应用。WebService能够在不同程序或者不同平台上开发的程序互相通信，是当前最成熟的SOA技术之一。Ajax是服务器端+客户端技术，能够实现页面的无刷新，通过简单案例，介绍了Ajax技术的实现方法。

6.6　习　　题

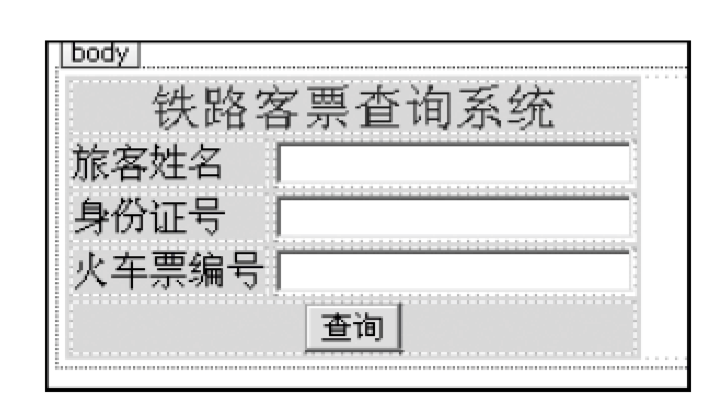
1. 简述Web Service的工作原理。
2. 简述Ajax所有技术元素的作用。
3. 简述使用XMLHttpRequest访问服务器的步骤。
4. 新建一个名为"DoctorForSearch"的网站来模拟总医院网站，新建一个名为"YSCX"的网站，来模拟各分院网站，通过WebService查询总医院医生门诊情况。

医生信息查询系统

请输入医生科室（支持模糊查询）　内科　查询

姓名	科室	职称
王丽	内科	高级
汪平	内科	中级

5. 实现铁路网上客票验证功能，输入旅客姓名、身份证号及火车票编号，单击"查询"按钮，提示验证结果，整个页面过程无刷新（提示：用Ajax来实现）。



第7章　数据导入、导出与报表

本章要点：

- 使用 GridView 读取 Excel 数据
- 将 Excel 表格数据写入 SQL Server 数据库
- 将 GridView 表格数据导出另存为 Excel 表格
- 将 GridView 表格数据生成固定格式的 Excel 报表

技能目标：

- 会编码获取 Excel 文件数据并展示在 GridView 表格中
- 会编写 GridView 写入数据库的代码
- 会编码实现 GridView 中数据导出到 Excel 文件中
- 会编写 GridView 表格数据生成固定格式报表的代码

7.1　项目导入

【项目场景】

在大学生成长记平台中，有如图 7.1 的数据表格，包括学生表、成绩表和荣誉表，分别记录学生基本信息、课程成绩和各项荣誉称号，在录入学生信息时，经常需要将整个班级学生信息整体导入学生信息表，如图 7.2 和图 7.3 所示。在信息查询的时候，用户需要将查询结果另存为 Excel 文件，如图 7.4 所示；整个系统需要打印包括基本信息、成绩、荣誉在内的学生成长记录汇总表，如图 7.5 和图 7.6 所示。

请结合项目功能和图 7.1～图 7.6，实现以下操作：

(1) 将学生信息按照班级整体导入；

(2) 将查询到的学生信息另存为自动格式的 Excel；

(3) 实现打印每个学生成长记录信息汇总表(固定格式的 Excel 表格)。

【问题引导】

(1) ASP.NET 如何读取 Excel。

(2) 如何将 Excel 表格数据导入数据库并展示在页面。

(3) 如何成批导出(另存)查询出来的 GridView 中的数据。

ADMIN-PC.stuDB - dbo.score

列名	数据类型	允许 Null 值
stuNo	varchar(30)	☑
term	varchar(50)	☑
course	varchar(50)	☑
score1	int	☑
remarks	varchar(MAX)	☑
		☐

ADMIN-PC.stuDB - dbo.honor

列名	数据类型	允许 Null 值
stuNo	varchar(30)	☑
honor1	varchar(50)	☑
gotDate	date	☑
grantingUnit	varchar(50)	☑
grade	varchar(50)	☑
		☐

ADMIN-PC.stuDB - dbo.student

列名	数据类型	允许 Null 值
stuNO	varchar(15)	☑
stuName	varchar(30)	☑
stuGender	varchar(6)	☑
birthDate	date	☑
profession	varchar(50)	☑
class1	varchar(50)	☑
department	varchar(50)	☑
instructor	varchar(30)	☑
countrySid	varchar(20)	☑
address	varchar(MAX)	☑
zipCode	varchar(50)	☑
familyTel	varchar(15)	☑
passWord	varchar(50)	☑
		☐

图 7.1　学生表、成绩表和荣誉表

	A	B	C	D	E	F	G	H	I	J	K	L	M
1	stuNO	stuName	stuGender	birthDate	professic	class1	departmer	instructc	countrySi	address	zipCode	familyTel	passWord
2	22	包晶晶	女	1995/5/1	软件技术	全日制201	软件与服务	汪园超	江苏	太仓	215411	110	123456
3	2	蔡政委	男	1995/5/2	软件技术	全日制201	软件与服务	汪园超	江苏	太仓	215411	110	123456
4	3	陈杭	男	1995/5/3	软件技术	全日制201	软件与服务	汪园超	江苏	太仓	215411	110	123456
5	4	丁道明	男	1995/5/4	软件技术	全日制201	软件与服务	汪园超	江苏	太仓	215411	110	123456
6	5	葛高文	女	1994/5/3	软件技术	全日制201	软件与服务	汪园超	江苏	太仓	215411	110	123456
7	6	顾丽丽	女	1995/5/6	软件技术	全日制201	软件与服务	汪园超	江苏	太仓	215411	110	123456
8	7	黄思强	男	1995/5/7	软件技术	全日制201	软件与服务	汪园超	江苏	太仓	215411	110	123456
9	8	姜越	女	1995/5/8	软件技术	全日制201	软件与服务	汪园超	江苏	太仓	215411	110	123456
10	9	金彬	男	1995/5/9	软件技术	全日制201	软件与服务	汪园超	江苏	太仓	215411	110	123456
11	10	刘长伟	男	########	软件技术	全日制201	软件与服务	汪园超	江苏	太仓	215411	110	123456
12	11	刘家驰	男	########	软件技术	全日制201	软件与服务	汪园超	江苏	太仓	215411	110	123456
13	12	刘毅	男	########	软件技术	全日制201	软件与服务	汪园超	江苏	太仓	215411	110	123456
14	13	秦江炜	女	########	软件技术	全日制201	软件与服务	汪园超	江苏	太仓	215411	110	123456
15	14	孙政	男	########	软件技术	全日制201	软件与服务	汪园超	江苏	太仓	215411	110	123456
16	15	汤鹤鸣	男	########	软件技术	全日制201	软件与服务	汪园超	江苏	太仓	215411	110	123456
17	16	陶银	女	1995/3/5	软件技术	全日制201	软件与服务	汪园超	江苏	太仓	215411	110	123456
18	17	王蕾	女	########	软件技术	全日制201	软件与服务	汪园超	江苏	太仓	215411	110	123456
19	18	夏彤彤	女	########	软件技术	全日制201	软件与服务	汪园超	江苏	太仓	215411	110	123456
20	19	许源	女	########	软件技术	全日制201	软件与服务	汪园超	江苏	太仓	215411	110	123456
21	20	薛奎	男	########	软件技术	全日制201	软件与服务	汪园超	江苏	太仓	215411	110	123456
22	21	张焱晖	男	########	软件技术	全日制201	软件与服务	汪园超	江苏	太仓	215411	110	123456
23	22	张燕	女	########	软件技术	全日制201	软件与服务	汪园超	江苏	太仓	215411	110	123456
24	23	仲冠文	男	########	软件技术	全日制201	软件与服务	汪园超	江苏	太仓	215411	110	123456
25	24	周宇明	男	########	软件技术	全日制201	软件与服务	汪园超	江苏	太仓	215411	110	123456
26	25	周韡	女	########	软件技术	全日制201	软件与服务	汪园超	江苏	太仓	215411	110	123456
27													

图 7.2　导入前的班级学生信息(Excel 表格)



学号	姓名	性别	出生日期	专业	班级	院系	辅导员	家乡	地址	邮编	家庭电话	密码	
1	包晶晶	女	1995/5/1 0:00:00	软件技术	全日制2013软件技术1班	软件与服务外包学院	汪园超	江苏宿迁	江苏宿迁北京路1号	215411	110	123456	☐
2	蔡政委	男	1995/5/2 0:00:00	软件技术	全日制2013软件技术1班	软件与服务外包学院	汪园超	江苏盐城	江苏盐城南京路1号	215411	110	123456	☐
3	陈杭	男	1995/5/3 0:00:00	软件技术	全日制2013软件技术1班	软件与服务外包学院	汪园超	江苏	太仓	215411	110	123456	☐
4	丁道明	男	1995/5/4 0:00:00	软件技术	全日制2013软件技术1班	软件与服务外包学院	汪园超	江苏	太仓	215411	110	123456	☐
5	葛高文	女	1994/5/3 0:00:00	软件技术	全日制2013软件技术1班	软件与服务外包学院	汪园超	江苏	太仓	215411	110	123456	☐
6	顾丽丽	女	1995/5/6 0:00:00	软件技术	全日制2013软件技术1班	软件与服务外包学院	汪园超	江苏	太仓	215411	110	123456	☐
7	黄思强	男	1995/5/7 0:00:00	软件技术	全日制2013软件技术1班	软件与服务外包学院	汪园超	江苏	太仓	215411	110	123456	☐

图 7.3　单击“批量导入”后的界面效果

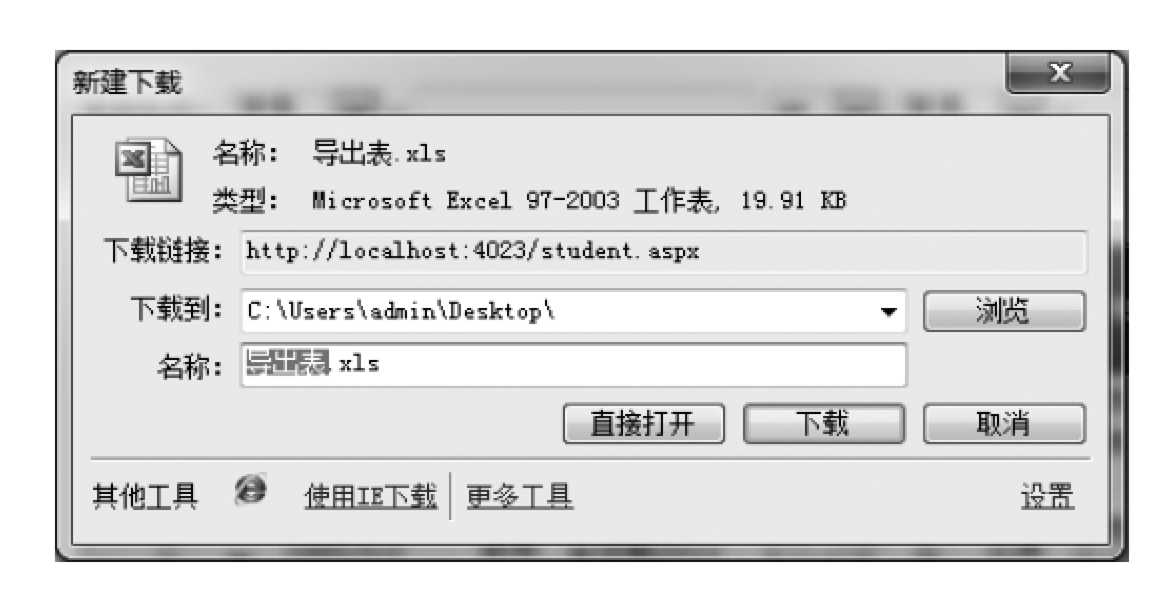


图 7.4-1　单击“导出”出现“另存为”对话框

	A	B	C	D	E	F	G	H	I	J	K	L	M
1	stuNO	stuName	stuGender	birthDate	professic	class1	departmer	instructc	countrySi	address	zipCode	familyTel	passWord
2	22	包晶晶	女	1995/5/1	软件技术	全日制201	软件与服务	汪园超	江苏	太仓	215411	110	123456
3	2	蔡政委	男	1995/5/2	软件技术	全日制201	软件与服务	汪园超	江苏	太仓	215411	110	123456
4	3	陈杭	男	1995/5/3	软件技术	全日制201	软件与服务	汪园超	江苏	太仓	215411	110	123456
5	4	丁道明	男	1995/5/4	软件技术	全日制201	软件与服务	汪园超	江苏	太仓	215411	110	123456
6	5	葛高文	女	1994/5/3	软件技术	全日制201	软件与服务	汪园超	江苏	太仓	215411	110	123456
7	6	顾丽丽	女	1995/5/6	软件技术	全日制201	软件与服务	汪园超	江苏	太仓	215411	110	123456
8	7	黄思强	男	1995/5/7	软件技术	全日制201	软件与服务	汪园超	江苏	太仓	215411	110	123456
9	8	姜越	女	1995/5/8	软件技术	全日制201	软件与服务	汪园超	江苏	太仓	215411	110	123456
10	9	金彬	男	1995/5/9	软件技术	全日制201	软件与服务	汪园超	江苏	太仓	215411	110	123456
11	10	刘长伟	男	########	软件技术	全日制201	软件与服务	汪园超	江苏	太仓	215411	110	123456
12	11	刘家驰	男	########	软件技术	全日制201	软件与服务	汪园超	江苏	太仓	215411	110	123456
13	12	刘毅	男	########	软件技术	全日制201	软件与服务	汪园超	江苏	太仓	215411	110	123456
14	13	秦江炜	女	########	软件技术	全日制201	软件与服务	汪园超	江苏	太仓	215411	110	123456
15	14	孙政	男	########	软件技术	全日制201	软件与服务	汪园超	江苏	太仓	215411	110	123456
16	15	汤鹤鸣	男	########	软件技术	全日制201	软件与服务	汪园超	江苏	太仓	215411	110	123456
17	16	陶银	女	1995/3/5	软件技术	全日制201	软件与服务	汪园超	江苏	太仓	215411	110	123456
18	17	王蕾	女	########	软件技术	全日制201	软件与服务	汪园超	江苏	太仓	215411	110	123456
19	18	夏彤彤	女	########	软件技术	全日制201	软件与服务	汪园超	江苏	太仓	215411	110	123456
20	19	许源	女	########	软件技术	全日制201	软件与服务	汪园超	江苏	太仓	215411	110	123456
21	20	薛奎	男	########	软件技术	全日制201	软件与服务	汪园超	江苏	太仓	215411	110	123456
22	21	张焱晖	男	########	软件技术	全日制201	软件与服务	汪园超	江苏	太仓	215411	110	123456
23	22	张燕	女	########	软件技术	全日制201	软件与服务	汪园超	江苏	太仓	215411	110	123456
24	23	仲冠文	男	########	软件技术	全日制201	软件与服务	汪园超	江苏	太仓	215411	110	123456
25	24	周宇明	男	########	软件技术	全日制201	软件与服务	汪园超	江苏	太仓	215411	110	123456
26	25	周赟	女	########	软件技术	全日制201	软件与服务	汪园超	江苏	太仓	215411	110	123456
27													

图 7.4-2　数据导出另存的效果

学号：1 查询 打印报表

学号	姓名	性别	出生日期	专业	班级	院系	籍贯	地址	邮编	家庭电话
1	包晶晶	女	1995/5/1 0:00:00	软件技术	全日制2013软件技术1班	软件与服务外包学院	江苏宿迁	江苏宿迁北京路1号	215411	110

课程名	成绩	学期
社交技术	88	2014-2015-1
C#程序设计	90	2014-2015-1
英语	92	2014-2015-2
和谐社会	96	2014-2015-2
数学	88	2015-1-1
三个代表	98	2015-4-5
体育	98	2015-6-7
ASP.NET程序设计	89	2015-6-7
JSP动态WEB开发	86	2016-7-8
PHP程序设计	98	2014-3-4

获奖	获得时间	级别	授予单位
红旗手	2015/5/6 0:00:00	市级	太仓工会
三号学生	2015/6/1 0:00:00	院级	学院
优秀干部	2014/2/1 0:00:00	院级	学院
优秀党员	2014/3/1 0:00:00	省级	江苏省教育厅
最佳歌手	2015/6/7 0:00:00	市级	太仓市文广局
一等奖学金	2014/5/6 0:00:00	省级	教育厅
国家励志奖学金	2015/6/6 0:00:00	国家级	国家教育部
助学金	2015/7/6 0:00:00	省级	教育厅

图 7.5　打印固定报表前的界面查询到某个学生的数据

大学生成长记录信息汇总表

学号		姓名		性别		出生日期		专业		班级		院系	
籍贯		地址						邮编		家庭电话			班主任评价
学习成绩	课程	成绩	学期	课程	成绩	学期	课程	成绩	学期	课程	成绩	学期	
													院系评价
主要荣誉和成果	荣誉名称	获奖时间	授予单位	级别	荣誉名称	获奖时间	授予单位	级别	荣誉名称	获奖时间	授予单位	级别	
													学院评定

图 7.6-1　打印前的固定表格

学生成绩获奖汇总表

学号	1	姓名	包品品	性别	女	出生日期	1995/5/1	专业	软件技术	班级	全日制2013 软件技术1班	院系	软件与服务外包学院
籍贯	江苏宿迁	地址	江苏宿迁北京路1号					邮编	215411	家庭电话	110		班主任评价
学习成绩	课程	成绩	学期	课程	成绩	学期	课程	成绩	学期	课程	成绩	学期	
	社交技术	88	2014-2015-1	ASP.NET程序设计	89	2015/6/7							
	C#程序设计	90	2014-2015-1	JSP动态WEB开发	86	2016/7/8							
	英语	92	2014-2015-2	PHP程序设计	98	2014/3/4							
	和谐社会	96	2014-2015-2										
	数学	88	2015/1/1										院系评价
	三个代表	98	2015/4/5										
	体育	98	2015/6/7										
主要荣誉和成果	荣誉名称	获奖时间	授予单位	级别	荣誉名称	获奖时间	授予单位	级别	荣誉名称	获奖时间	授予单位	级别	
	红旗手	2015/5/6	市级	太仓工会	国家励志奖学金	2015/6/6	国家级	国家教育部					
	三号学生	2015/6/1	院级	学院	助学金	2015/7/6	省级	教育厅					
	优秀干部	2014/2/1	院级	学院									学院评定
	优秀党员	2014/3/1	省级	江苏省教育厅									
	最佳歌手	2015/6/7	市级	太仓市文广局									
	一等奖学金	2014/5/6	省级	教育厅									

图 7.6-2　打印后数据报表效果

（4）如何将界面 GridView 中的数据直接打印成固定格式的报表。

7.2　技术与知识准备

7.2.1　读取 Excel 数据并显示在 GridView 控件中

【示例 7.1】　有一个 Excel 表格 ch7_1，数据如图 7.7(a)所示，编写代码，将表格数据读入界面的 GridView1 中，运行结果如图 7.7(b)所示。



图 7.7　将 Excel 数据读入 GridView1

【步骤 1】在界面添加一个按钮“导入”，添加一个“FileUpload1”控件和 GridView1，设计界面如图 7.8 所示。

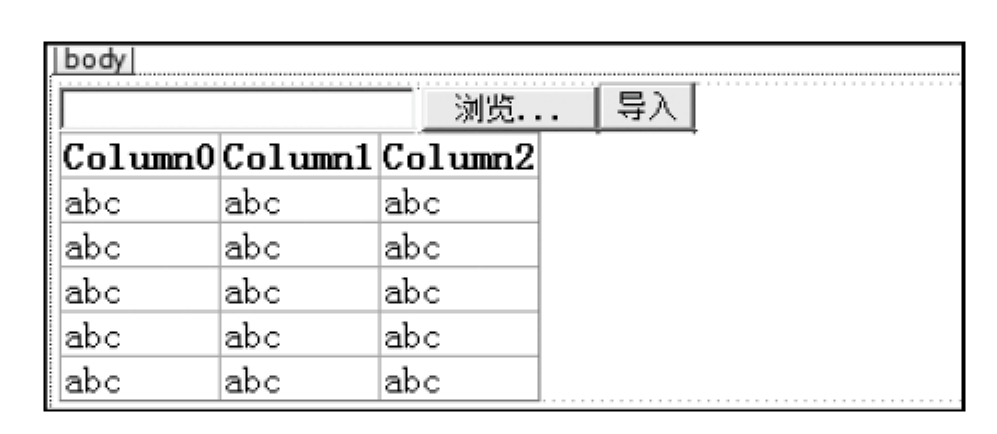


图 7.8　导入数据界面

【步骤 2】在 Button1-"导入"按钮编写导入 Excel 数据的代码。

```
using System.Data;
using System.Text;
using System.Data.SqlClient;
using System.Data.OleDb;
using System.Data.SqlClient;
//添加以上引用
  if(FileUpload1.PostedFile.FileName=="")
    { Response.Write("<script>alert('请您选择 Excel 文件')</script>");
      }
      else
      {
       string strconn="Provider=Microsoft.Jet.OLEDB.4.0;Data Source="+ FileUpload1.
       PostedFile.FileName.ToString()+";Extended Properties= Excel 8.0;";
       OleDbConnection mycn=new OleDbConnection(strconn);
       mycn.Open();
       DataSet grid=new DataSet();
       OleDbDataAdapter da=new OleDbDataAdapter("select * from [Sheet1$]",mycn);
       da.Fill(grid, "ExcelInfo");
       GridView1.DataSource=grid.Tables["ExcelInfo"].DefaultView;
      //添加一个 GridView2,作为数据缓冲,往数据库添加数据,不可见。visible定义为 false
       GridView1.DataBind();
       mycn.Close();
       mycn.Dispose();          }
```

7.2.2 将 GridView 表格中数据写入 SQL Server 数据表中

【示例 7.2】 将示例 7.1 中 GridView1 的数据写入 SQL Server 数据库。运行效果如图 7.9 所示。

ADMIN-PC.7-1 - dbo.7-1

	no	name	num	remark
▶	1	茄子	3	NULL
	2	黄瓜	4	NULL
	3	丝瓜	5	NULL
*	NULL	NULL	NULL	NULL

图 7.9 导入到 SQL Server 数据库后的效果

【步骤 1】在界面添加一个按钮"写入数据库按钮",如图 7.10 所示。
【步骤 2】在"写入数据库"按钮中编写代码。

```
protected void Button2_Click(object sender, EventArgs e)
{
string constring=ConfigurationManager.ConnectionStrings["ch07"].ToString();
                                                  //定义连接数据库的字符串
```

body

浏览... 导入 写入数据库

Column0	Column1	Column2
abc	abc	abc
abc	abc	abc
abc	abc	abc
abc	abc	abc
abc	abc	abc

图 7.10 “写入数据库”按钮

```
foreach(GridViewRow rowview in GridView1.Rows)
 {
  SqlConnection conn=new SqlConnection(constring);
  string sql=string.Format("insert into ch07 values('{0}','{1}','{2}','{3}')", rowview.
  Cells[0].Text, rowview.Cells[1].Text,rowview.Cells[2].Text,rowview.Cells[3].Text);
  SqlCommand cmd=new SqlCommand(sql, conn);
  conn.Open();
  cmd.ExecuteNonQuery();
  conn.Close();
  conn.Dispose();
  }
 }
```

7.2.3 将 Excel 数据导出(另存为)Excel

1. 将整个 html 全部输出 Excel

此法将 html 中所有的内容,如按钮,表格,图片等全部输出到 Excel 中。

```
Response.Clear();
Response.Buffer=true;
    Response.AppendHeader("Content-Disposition","attachment;filename="+ DateTime.Now.
    ToString("yyyyMMdd")+".xls");
Response.ContentEncoding=System.Text.Encoding.UTF8;
Response.ContentType="application/vnd.ms-excel";
this.EnableViewState=false;
```

这里利用了 ContentType 属性,它默认的属性为 text/html,这时将输出为超文本,即常见的网页格式到客户端,如果改为 ms-excel 将输出 excel 格式,也就是说以电子表格的格式输出到客户端,这时浏览器将提示下载保存。ContentType 的属性还包括:image/JPEG;text/HTML;image/GIF;vnd.ms-excel/msword 。同理,也可以输出(导出)图片、Word 文档等。下面的方法,也都用了这个属性。

注意:此方法确实可以导出 Excel,但更改 ContentType 的属性依然导出 Excel,测试环境是 WPS Office 2010 个人版,搜狗浏览器。

2. 将 GridView 控件中的数据导出 Excel

上述方法虽然实现了导出的功能,但同时把按钮、分页框等 html 中的所有输出信息

导了进去。而一般要导出的是数据,GridView 控件上的数据。

```
System.Web.UI.Control ctl=this.GridView1;
//DataGrid1 在窗体中拖放的控件
HttpContext.Current.Response.AppendHeader("Content-Disposition","attachment;filename=
Excel.xls");
HttpContext.Current.Response.Charset="UTF-8";
HttpContext.Current.Response.ContentEncoding=System.Text.Encoding.Default;
HttpContext.Current.Response.ContentType="application/ms-excel";
ctl.Page.EnableViewState=false;
System.IO.StringWriter tw=new System.IO.StringWriter();
System.Web.UI.HtmlTextWriter hw=new System.Web.UI.HtmlTextWriter(tw);
ctl.RenderControl(hw);
HttpContext.Current.Response.Write(tw.ToString());
HttpContext.Current.Response.End();
```

如果 DataGrid 用了分页,导出的是当前页的信息,也就是 DataGrid 中显示的信息,而不是 select 语句的全部信息。

注意:此法总是报错,提示 gridview 应该包含在 runat 标记内。

3. 将 DataSet 中的数据导出 Excel

有了上边的思路,就是将在导出的信息,输出(Response)客户端,这样就可以导出了。那么把 DataSet 中的数据导出,也就是把 DataSet 中的表中的各行信息,以 ms-excel 的格式 Response 到 http 流。说明:参数 ds 应为填充有数据表的 DataSet,文件名是全名,包括后缀名,如 Excel2006.xls。

```
public void CreateExcel(DataSet ds, string FileName)
{
    HttpResponse resp;
    resp=Page.Response;
    resp.ContentEncoding=System.Text.Encoding.GetEncoding("GB2312");
    resp.AppendHeader("Content-Disposition", "attachment;filename="+FileName+".xls");
    string colHeaders="", ls_item="";
    //定义表对象与行对象,同时用 DataSet 对其值进行初始化
    DataTable dt=ds.Tables[0];
    DataRow[] myRow=dt.Select();   //可以类似 dt.Select("id>10")的形式达到数据筛选目的
    int i=0;
    int cl=dt.Columns.Count;
   //取得数据表各列标题,各标题之间以 t 分割,最后一个列标题后加回车符
    for(i=0; i<cl; i++)
    {
       if(i==(cl-1))                               //最后一列,加 n
        {
          colHeaders+=dt.Columns[i].Caption.ToString()+"\n";
        }
```

```
            else
             {
              colHeaders+=dt.Columns[i].Caption.ToString()+"\t";
             }
             }
             resp.Write(colHeaders);
             //向 HTTP 输出流中写入取得的数据信息
             //逐行处理数据
             foreach(DataRow row in myRow)
             {
                 //当前行数据写入 HTTP 输出流,并且置空 ls_item 以便下行数据
                 for(i=0; i<cl; i++)
                 {
                     if(i==(cl-1))                     //最后一列,加 n
                     {
                         ls_item+=row[i].ToString()+"\n";
                     }
                     else
                     {
                         ls_item+=row[i].ToString()+"\t";
                     }
                 }
                 resp.Write(ls_item);
                 ls_item="";
             }
             resp.End();
         }
```

7.2.4 将 GridVirew 中数据生成固定格式 Excel 报表

将 GridView 表格数据导出到固定格式的 Excel 文件,首先创建一个 Excel 格式表格,然后编写代码将数据导出到固定格式的 Excel 表格中,这里的重点是如何确定固定表格中单元格进行读取,写入数据。

【示例 7.3】 有一个顶岗实习指导记录表,请根据数据库查询出来的 Gridview 数据,如图 7.11 所示,生成数据导入固定格式的报表,如图 7.12 所示。

Module_Guiding…	Module_Guiding…	Module_Guiding…	Module_Guiding…	Module_Guiding…	Module_Guiding…	Module_Guiding…	Module_Guiding…	Module_Guiding…	Module_Guiding…	Module_Guiding…	Module_Guiding…
5	郑广成	软件与服务外…	软件技术	软件0914	孔小兵	新世基(太仓)科…	2011-10-26 00:…	现场指导	NULL	ddddd	NULL
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

图 7.11　需要导出的数据信息

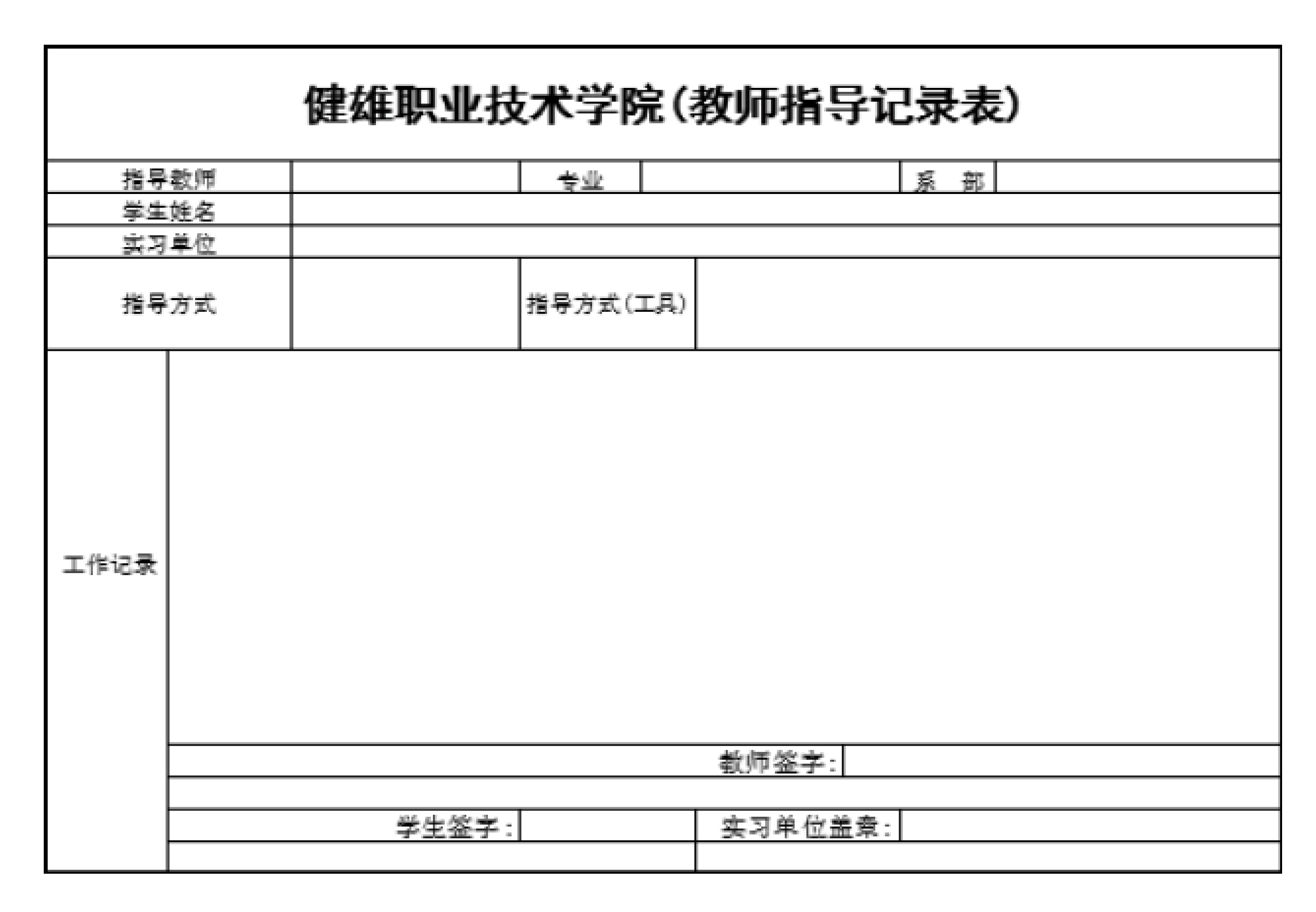


图 7.12　需要将数据导出到 Excel 报表格式

关键代码:

```
using Microsoft.Office.Interop.Excel;
public partial class Teacher_PracticeInformation_PrintGuidingRecord : System.Web.UI.Page
{
    #region 创建对象
    RightTeacherSvc teachersvc=new RightTeacherSvc();
    ModuleIntershipsSituationSvc intersvc=new ModuleIntershipsSituationSvc();
    ModuleGuidingRecordSvc guidingsvc=new ModuleGuidingRecordSvc();
    ModuleGuidingRecordEntity guentity=new ModuleGuidingRecordEntity();
    #endregion
    protected void Page_Load(object sender, EventArgs e)
    {
        if(!Page.IsPostBack)
        {
        int ID=Convert.ToInt32(Request.QueryString["strNO"].ToString().Trim());
        guentity.PModuleGuidingID=ID;
         TextBox1.Text=guidingsvc.GetListByEntity(guentity)[0].
         PModuleGuidingDepartmentName;
            TextBox2.Text=guidingsvc.GetListByEntity(guentity)[0].
            PModuleGuidingProfessionalName;
            TextBox3.Text=guidingsvc.GetListByEntity(guentity)[0].
            PModuleGuidingClassName;
            TextBox4.Text=guidingsvc.GetListByEntity(guentity)[0].
            PModuleGuidingEnterpriseName;
```

```
            TextBox5.Text=guidingsvc.GetListByEntity(guentity)[0].
            PModuleGuidingWayContent;
            TextBox6.Text=guidingsvc.GetListByEntity(guentity)[0].
            PModuleGuidingStudentName;
            TextBox7.Text=guidingsvc.GetListByEntity(guentity)[0].
            PModuleGuidingWay;
            TextBox8.Text=guidingsvc.GetListByEntity(guentity)[0].
            PModuleGuidingContent;
        }
    }
    protected void Button1_Click(object sender, EventArgs e)
    {
        ApplicationClass xls=new ApplicationClass();
        Workbook mybook=null;
        Worksheet mysheet=null;
        object omissing=System.Reflection.Missing.Value;
        //打开指定好的 Excel 工作簿
        xls.Workbooks._Open(Server.MapPath("../../Bin")+"\\"+"教师指导记录表.xls",
        omissing, omissing, omissing, omissing, omissing, omissing, omissing, omissing,
        omissing, omissing, omissing, omissing);
                                                  //12个 omissing
        mybook=xls.Workbooks[1];
        xls.Visible=true;
        mysheet=(Worksheet)mybook.ActiveSheet;
        //
        mysheet.Cells[3, 3]=Session["TeacherName"].ToString();
        //
        mysheet.Cells[3, 5]=TextBox2.Text;
        //
        mysheet.Cells[3, 9]=TextBox1.Text;
        //
        mysheet.Cells[4, 3]=TextBox6.Text;
        //
        mysheet.Cells[5, 3]=TextBox4.Text;
        //
        mysheet.Cells[6, 3]=TextBox7.Text;
        //
        mysheet.Cells[6, 6]=TextBox5.Text;
        //
        mysheet.Cells[12, 2]=TextBox8.Text;
        //
        mysheet.Cells[6, 3]=TextBox7.Text;
        //指导教师签字
        mysheet.Cells[31, 7]=Session["TeacherName"].ToString();
```

```
        //添加日期
        mysheet.Cells[32, 2] = DateTime.Now.Year + "年" + DateTime.Now.Month + "月" +
        DateTime.Now.Day+"日";
        //
        mysheet.Cells[33, 4]=TextBox6.Text;
        //添加日期
        mysheet.Cells[34, 2] = DateTime.Now.Year + "年" + DateTime.Now.Month + "月" +
        DateTime.Now.Day+"日";
        //
        mysheet.Cells[33, 8]=TextBox4.Text;
        //添加日期
        mysheet.Cells[34, 6] = DateTime.Now.Year + "年" + DateTime.Now.Month + "月" +
        DateTime.Now.Day+"日";
    }
    protected void Button2_Click(object sender, EventArgs e)
    {
        Response.Redirect("GuidingRecordOut.aspx");
    }
```

7.3 项目训练

通过对以上内容的学习，对于 ASP. NET 操作 Excel 进行数据导入导出和报表的主要方法有了一定的熟悉和掌握，接下来要完成课内训练项目，实现如图 7.1～图 7.6 所示效果。

项目任务 1：将学生信息按照班级整体导入。

【步骤 1】根据提供的素材，在界面添加“成批导入按钮”，添加 Gridview2，需要说明的是 GridView2 用来读取 excel 文件数据，然后将 Gridview2 数据填入数据库，在通过 excel 读取数据库数据显示，表示导入成功。

备注：素材中需要导入的文件为“sysstu. xls”。

【步骤 2】编写“成批导入”按钮的关键代码。

```
using System.Data.OleDb;
using System.Data.SqlClient;
protected void Button4_Click(object sender, EventArgs e)
        {
            try
            {
                if(FileUpload1.PostedFile.FileName=="")
                {
                    Response.Write("<script>alert('请您选择 Excel 文件')
                    </script>");
                }
```

```
            else
            {
                string strconn="Provider=Microsoft.Jet.OLEDB.4.0;Data Source="+
               FileUpload1.PostedFile.FileName.ToString()+";Extended Properties=
               Excel 8.0;";
                OleDbConnection mycn=new OleDbConnection(strconn);
                mycn.Open();
                DataSet grid=new DataSet();
                OleDbDataAdapter da=new OleDbDataAdapter("select * from [sheet1
                $]", mycn);
                da.Fill(grid, "ExcelInfo");
                GridView2.DataSource=grid.Tables["ExcelInfo"].DefaultView;
    //添加一个 GridView2,作为数据缓冲,往数据库添加数据,不可见。visible定义为 false
                GridView2.DataBind();
                mycn.Close();
                mycn.Dispose();
                Student student=new Student();
                StudentManager studentManager=new StudentManager();
                foreach(GridViewRow rowview in GridView2.Rows)
                {
//在这里说明是该源程序使用的是面向对象的方法,故在此引用对象,对象所属类在此直接引用
//即可。
                    student.StuNO=rowview.Cells[0].Text;
                    student.StuName=rowview.Cells[1].Text;
                    student.StuGender=rowview.Cells[2].Text;
                    student.BirthDate=rowview.Cells[3].Text;
                    student.Profession=rowview.Cells[4].Text;
                    student.Class1=rowview.Cells[5].Text;
                    student.Department=rowview.Cells[6].Text;
                    student.Instructor=rowview.Cells[7].Text;
                    student.CountrySid=rowview.Cells[8].Text;
                    student.Address=rowview.Cells[9].Text;
                    student.ZipCode=rowview.Cells[10].Text;
                    student.FamilyTel=rowview.Cells[11].Text;
                    student.PassWord=rowview.Cells[12].Text;
                    studentManager.addStudent(student);
                }
                refresh();
                Response.Write("<script>alert('导入成功!');</script>");
            }
        }
        catch(Exception ex)
        {
            Response.Write("<script>alert('导入失败,导入的数据必须是 Excel 格式!
```

```
            ');</script>");
        }
        finally
        {

        }
    }
```

项目任务 2：将查询到的学生信息另存为自动格式的 Excel。

打开素材站程序“studentSYS-课内主项目”，编写导出按钮的关键代码：

```
protected void Button2_Click(object sender, EventArgs e)
    {
        Microsoft.Office.Interop.Excel.Application excel = new Microsoft.Office.
        Interop.Excel.Application();
        if(this.GridView1.Rows.Count==0)
        {
            Response.Write("<script>alert('没有查找到数据,无法导出!')");
        }
        else
        {
            this.GridView1.AllowPaging=false;
                                    //将有分页的 GridView 中的数据全部导出到 Excel
            export("application/ms-excel", "导出表.xls");
            //换成 export("application/ms-word", "工作人员.doc"); 那么导出的
            //就是 Word 格式的了
            this.GridView1.AllowPaging=true;
        }
    }
    public void export(string FileType, string FileName)
    {
        string style=@"<style>.text{mso-number-format:@}</script>";
                                    //导入到 Excel 时,保存表里数字列中前面存在的 0
        //PrepareGridViewForExport(Control gv)     //将模板列显示出来
        Response.Clear();
        Response.Charset="GB2312";
        Response.ContentEncoding=Encoding.UTF7;
        Response.AppendHeader("Content-Disposition", "attachment;filename="+
       HttpUtility.UrlEncode(FileName, Encoding.UTF8).ToString());
        Response.ContentType=FileType;
        this.EnableViewState=false;
        this.GridView1.AllowPaging=false;
        System.Globalization.CultureInfo myCItrad=new System.
        Globalization.CultureInfo("ZH-CN", true);
```

```
            StringWriter sw=new StringWriter();
            HtmlTextWriter htw=new HtmlTextWriter(sw);
            this.GridView1.RenderControl(htw);
            Response.Write(style);
            Response.Write(sw.ToString());
            //Response.Write(dt.ToString());
            Response.End();
        }
```

项目任务3：实现打印每个学生成长记录信息汇总表(固定格式的 Excel 表格)。

【步骤1】制作固定格式 Excel 报表模板，如图7.13所示。

学生成绩获奖汇总表

学号		姓名		性别		出生日期		专业		班级		院系	
籍贯		地址						邮编		家庭电话			班主任评价
学习成绩	课程	成绩	学期	课程	成绩	学期	课程	成绩	学期	课程	成绩	学期	
													院系评价
主要荣誉和成果	荣誉名称	获奖时间	授予单位	级别	荣誉名称	获奖时间	授予单位	级别	荣誉名称	获奖时间	授予单位	级别	
													学院评定

图7.13 报表模板

【步骤2】打开素材程序系统“studentSYS-课内主项目”，附加好数据库“stuDB”数据库。

【步骤3】编写 personUI.aspx 页面中“打印报表”按钮的关键代码。

```
using Microsoft.Office.Interop.Excel;
        protected void Button1_Click(object sender, EventArgs e)
        {
            Application xls=new Application();
            Workbook mybook=null;
            Worksheet mysheeet=null;
            object omissing=System.Reflection.Missing.Value;
            xls.Workbooks.Open("G:\\stuall.xls", omissing, omissing, omissing, omissing,
            omissing, omissing, omissing, omissing, omissing,
            omissing, omissing, omissing);            //Server.MapPath("../bin")+"\\"+
            mybook=xls.Workbooks[1];                  //c:\\李曼\张圣诞\\stu.xls
            xls.Visible=true;
```

```
mysheeet= (Worksheet)mybook.ActiveSheet;
///将学生基本信息按照表格格式填写///
mysheeet.Cells[2, 2]=GridView1.Rows[0].Cells[0].Text;
mysheeet.Cells[2, 4]=GridView1.Rows[0].Cells[1].Text;
mysheeet.Cells[2, 6]=GridView1.Rows[0].Cells[2].Text;
mysheeet.Cells[2, 8]=GridView1.Rows[0].Cells[3].Text;
mysheeet.Cells[2, 10]=GridView1.Rows[0].Cells[4].Text;
mysheeet.Cells[2, 12]=GridView1.Rows[0].Cells[5].Text;
mysheeet.Cells[2, 14]=GridView1.Rows[0].Cells[6].Text;
mysheeet.Cells[3, 2]=GridView1.Rows[0].Cells[7].Text;
mysheeet.Cells[3, 4]=GridView1.Rows[0].Cells[8].Text;
mysheeet.Cells[3, 10]=GridView1.Rows[0].Cells[9].Text;
mysheeet.Cells[3, 12]=GridView1.Rows[0].Cells[10].Text;
///学生成绩的填写///
foreach(GridViewRow row in GridView2.Rows)
{
    int i=row.RowIndex;
    if(row.RowIndex <7)
    {
        mysheeet.Cells[i+5, 2]=GridView2.Rows[i].Cells[0].Text;
        mysheeet.Cells[i+5, 3]=GridView2.Rows[i].Cells[1].Text;
        mysheeet.Cells[i+5, 4]=GridView2.Rows[i].Cells[2].Text;
    }
    if(row.RowIndex >=7)
    {
        mysheeet.Cells[i+5-7, 5]=GridView2.Rows[i].Cells[0].Text;
        mysheeet.Cells[i+5-7, 6]=GridView2.Rows[i].Cells[1].Text;
        mysheeet.Cells[i+5-7, 7]=GridView2.Rows[i].Cells[2].Text;
    }
    if(row.RowIndex >=14)
    {
        mysheeet.Cells[i+5-14, 8]=GridView2.Rows[i].Cells[0].Text;
        mysheeet.Cells[i+5-14, 9]=GridView2.Rows[i].Cells[1].Text;
        mysheeet.Cells[i+5-14, 10]=GridView2.Rows[i].Cells[2].Text;
    }
    if(row.RowIndex >=21)
    {
        mysheeet.Cells[i+5-21, 11]=GridView2.Rows[i].Cells[0].Text;
        mysheeet.Cells[i+5-21, 12]=GridView2.Rows[i].Cells[1].Text;
        mysheeet.Cells[i+5-21, 13]=GridView2.Rows[i].Cells[2].Text;
    }
}
///学生荣誉整体写入///
foreach(GridViewRow row in GridView3.Rows)
{
    int i=row.RowIndex;
```

```
            int k=i+13;
            if(row.RowIndex < 6)
            {
                mysheeet.Cells[k, 2]=GridView3.Rows[i].Cells[0].Text;
                mysheeet.Cells[k, 3]=GridView3.Rows[i].Cells[1].Text;
                mysheeet.Cells[k, 4]=GridView3.Rows[i].Cells[2].Text;
                mysheeet.Cells[k, 5]=GridView3.Rows[i].Cells[3].Text;
            }
            if(row.RowIndex >=6)
            {
                mysheeet.Cells[k-6, 6]=GridView3.Rows[i].Cells[0].Text;
                mysheeet.Cells[k-6, 7]=GridView3.Rows[i].Cells[1].Text;
                mysheeet.Cells[k-6, 8]=GridView3.Rows[i].Cells[2].Text;
                mysheeet.Cells[k-6, 9]=GridView3.Rows[i].Cells[3].Text;
            }
            if(row.RowIndex >=12)
            {
                mysheeet.Cells[k-12, 10]=GridView3.Rows[i].Cells[0].Text;
                mysheeet.Cells[k-12, 11]=GridView3.Rows[i].Cells[1].Text;
                mysheeet.Cells[k-12, 12]=GridView3.Rows[i].Cells[2].Text;
                mysheeet.Cells[k-12, 13]=GridView3.Rows[i].Cells[3].Text;
            }
        }
```

7.4 平行项目训练

1. 训练内容

在 TeamManager 系统中,后台数据库"TeamDB"中有考核记录表"ItemScoreSet",现在需要通过界面"TeamInput1.aspx"进行数据整体导入,待导入的 excel 表格的格式与数据库表格一致。请编写代码实现 Excel 数据的成批导入。

同样在该项目,打开源程序,在界面"TeamSeek.aspx"中实现数据导出功能。

程序相关效果如图 7.14~图 7.16 所示。

ADMIN-PC.TeamDB - dbo.ItemScoreSet | ADMIN-PC.Tea

列名	数据类型	允许 Null 值
TeamItems	varchar(50)	☑
ItemType	varchar(50)	☑
StandardScore	float	☑
FullMarks	int	☑
SingleScore	int	☑
grade	varchar(50)	☑
IsAddScore	char(2)	☑
		☐

图 7.14 数据库表格

考核分类		Column0	Column1	Column2
项目类型		abc	abc	abc
达标分		abc	abc	abc
满分		abc	abc	abc
个人得分		abc	abc	abc
成绩		abc	abc	abc
加分项				
确定 取消 导入数据				

图 7.15　导入数据的界面效果(具体见源程序素材)

td.style39 ms　并且　TeamItems　模糊　删除　查询　导出报表

考核项目	考核类型	达标分数	总分	得分	等级	是否加分	□全选
数据绑定	数据绑定	数据绑定	数据绑定	数据绑定	数据绑定	数据绑定	□
数据绑定	数据绑定	数据绑定	数据绑定	数据绑定	数据绑定	数据绑定	□
数据绑定	数据绑定	数据绑定	数据绑定	数据绑定	数据绑定	数据绑定	□
数据绑定	数据绑定	数据绑定	数据绑定	数据绑定	数据绑定	数据绑定	□
数据绑定	数据绑定	数据绑定	数据绑定	数据绑定	数据绑定	数据绑定	□

图 7.16　需要导出数据的查询界面

2. 训练目的

进一步训练并巩固学生对 Excel 数据导入导出设计与编码能力。

3. 训练过程

【步骤 1】打开素材源程序,在“TeamInput1.aspx”界面编写“导入按钮”。

```
using System.Data.OleDb;
using System.Data.SqlClient;
ItemScoreSet sst=new ItemScoreSet();
List<ItemScoreSet> isss=new List<ItemScoreSet>();
protected void Button3_Click(object sender, EventArgs e)
        {
            try
            {
                if(FileUpload1.PostedFile.FileName=="")
                {
                Response.Write("<script>alert('请您选择 Excel 文件')</script>");
                }
                else
                {
                 string strconn="Provider=Microsoft.Jet.OLEDB.4.0;Data Source="+
                FileUpload1.PostedFile.FileName.ToString()+";Extended Properties=
                Excel 8.0;";
                 OleDbConnection mycn=new OleDbConnection(strconn);
```

```
                    mycn.Open();
                    DataSet grid=new DataSet();
                    OleDbDataAdapter da=new OleDbDataAdapter("select * from
                    [sheet1$]", mycn);
                    da.Fill(grid, "ExcelInfo");
                    GridView2.DataSource=grid.Tables["ExcelInfo"].DefaultView;
         //添加一个 GridView2,作为数据缓冲,往数据库添加数据,不可见。visible定义为 false
                    GridView2.DataBind();
                    mycn.Close();
                    mycn.Dispose();
                    Student student=new Student();
                    StudentManager studentManager=new StudentManager();
                   foreach(GridViewRow rowview in GridView2.Rows)
                   {
//在这里说明是该源程序使用的是面向对象的方法,故在此引用对象,对象所属类在此直接引用
//即可,相关方法和集合已经定义好。
                   sst.TeamItems=rowview.Cells[0].Text;
                   sst.ItemType=rowview.Cells[1].Text;
                   sst.StandardScore=int.Parse(rowview.Cells[2].Text);
                   sst.FullMarks=int.Parse(rowview.Cells[3].Text);
                   sst.SingleScore=int.Parse(rowview.Cells[4].Text);
                   sst.Grade=rowview.Cells[5].Text;
                   sst.IsAddScore=int.Parse(rowview.Cells[6].Text);
                   isss.addItemScoreSet(student);
                       }
                       refresh();
                       Response.Write("<script>alert('导入成功!');</script>");
                   }
               }
               catch(Exception ex)
               {
                   Response.Write("<script>alert('导入失败,导入的数据必须是 Excel 格式!
                   ');</script>");
               }
               finally
               {

               }
           }
```

【步骤 2】打开素材源程序,在“TeamScore.aspx”界面编写“导出报表”。

方法 1:

```
protected void btnOutPut_Click(object sender, EventArgs e)
        {
```

```
        //导出数据的方法有 10 种左右。针对 Excel。
        //System.Web.UI.Control
        Control ctl=GridView1;
        HttpContext.Current. Response. AppendHeader ( " Content - Disposition ", "
        attachment;filename=excel1.xls");                    //默认名称
        HttpContext.Current.Response.Charset="UTF-8";
        HttpContext.Current.Response.ContentType="application/ms-excel";
                                                             //定义导出的类型
        HttpContext.Current.Response.ContentEncoding=System.Text.Encoding.Default;
                                                             //定义编码格式
        ctl.Page.EnableViewState=false;
        StringWriter tw=new StringWriter();                  //定义写对象
        HtmlTextWriter hw=new HtmlTextWriter(tw);             //定义要输出的页对象
        ctl.RenderControl(hw);
        HttpContext.Current.Response.Write(tw.ToString());   //写出内容
        HttpContext.Current.Response.End();                  //结束

        //网络咨询,谷歌、百度等。论坛-可以实现你开发游戏。
    }
```

方法 2:

```
    ///<summary>
    ///定义一个导出报表的方法(函数)。
    ///</summary>
    ///<param name="dgv"></param>
    ///<param name="tittle"></param>
    public void Export_DGV(GridView dgv)
    {
        if(dgv==null || dgv.Rows.Count<1 || dgv.Columns.Count<1)
        {
            Response.Write("<script>alert('没有数据!');</script>");
            return;
        }
        int colscount=dgv.Columns.Count;
                Application objExcel=new Application();
        Workbook objWorkbook=null;
        Worksheet objsheet=null;

        //try
        //{
            //申明对象
            if(objExcel==null)
            {
                Response.Write("<script>alert('无法创建 Excel 对象,可能您的机器未
                安装 Excel');</script>");
                return;
```

```
        }
        objWorkbook=objExcel.Workbooks.Add(true);
        objsheet=(Worksheet)objWorkbook.ActiveSheet;
                                      //当前工作表为活动状态,默认打开。
        //设置 Excel 不可见
        objExcel.Visible=true;


        for(int i=1; i<=dgv.Columns.Count;i++)
        {
            objExcel.Cells[1,i]=dgv.Columns[i-1].HeaderText.Trim();
       }

        //向 Excel 中逐行逐列写入表格中的数据
        for(int row=1; row<=dgv.Rows.Count; row++)
        {
            for(int col=1; col<=colscount; col++)
            {
                        objExcel.Cells[row+1, col]=dgv.Rows[row-1].Cells[col
                        -1].Text.ToString().Trim();
            }
        }

}
protected void Button1_Click1(object sender, EventArgs e)
{
    Export_DGV(GridView1);
}
```

7.5 总　　结

本章介绍了 ASP.NET 对 Excel 表格的处理技术,重点讲述和训练了 Excel 数据导入 SQl Servevr 数据库、将数据库数据导出到 Excel、生成固定格式的 Excel 报表等方法,尤其是 Excel 数据导入与导出使用了 GridView 控件作为载体,既体现了数据操作的可见性,更提高了操作可理解性,提高了数据维护的效率。

7.6 习　　题

1. 编写将工作表 Sheet1 的内容读取到 DataSet 的关键代码。
2. 写出将 GridView 数据直接生成 Excel 文件的关键代码。
3. 使用 Excel 报表和数据导入导出有什么好处?

第 8 章　系统优化与测试

本章要点：

- 系统的优化
- 系统的测试

技能目标：

- 会进行系统优化
- 会利用 VS 自带单元测试进行测试

8.1　项目导入

【项目场景】

某公司要开发一个员工查询系统，根据员工部门名称，查询员工详细信息，请你为该公司开发一个系统，实现信息查询功能，用存储过程实现，界面如图 8.1 所示。



图 8.1　查询系统

【问题引导】

(1) 系统如何进行优化。
(2) 系统如何进行测试。

8.2　技术与知识准备

8.2.1　使用 vs. net 平台优化调试系统

一个系统建好并准备发布前，经过优化，系统运行速度会有很大提高，在访问人数较多时不至于出现服务器压力过大的情况。

1. 程序代码优化

(1) 对连接字符串的优化。

String 类对象是不可改变的，对于 String 对象的重新赋值在本质上是重新创建了一个 String 对象并将新值赋予该对象，其方法 ToString 对性能的提高并非很显著。在处理

字符串时，最好使用 StringBuilder 类，其.NET 命名空间是 System.Text。该类并非创建新的对象，而是通过 Append，Remove，Insert 等方法直接对字符串进行操作，通过 ToString 方法返回操作结果。

例如：

```
Using System.Text;
StringBuilder str=new StringBuilder();
str.Append("太仓");
str.Append("欢迎你");
Response.Write(str.ToString);                //显示操作结果
```

（2）当保存多个信息时，尽量用对象。

例如，当要将用户名、用户密码、用户类别保存时，尽量不要用三个 Session 来保存，而是定义一个类，包含用户名、密码和类别三个属性，实例化为对象，分别将用户输入的用户名、密码和类别赋值给该对象的三个属性，然后将该对象赋值给 Session，代码如下：

```
tbUse us=new tbUse();
us.Name=txtName.Text;
us.Pwd=txtPwd.Text;
us.Type=txtType.Text;
Session["User"]=us;
```

（3）使用 IsPostBack。

网页要避免不必要的服务器往返过程，添加 IsPostBack 可以判断是否让有关代码只在第一次加载时运行，这在一定程度上可以提高页面运行性能。

（4）使用 Ajax。

AJAX 即“Asynchronous Javascript And XML”（异步 JavaScript 和 XML），是指一种创建交互式网页应用的网页开发技术。通过在后台与服务器进行少量数据交换，AJAX 可以使网页实现异步更新。这意味着可以在不重新加载整个网页的情况下，对网页的某部分进行更新。

2. 数据操作优化

（1）缓存技术。

缓存功能是大型网站设计一个很重要的部分。由数据库驱动的 Web 应用程序，如果需要改善其性能，最好的方法是使用缓存功能。可能的情况下尽量使用缓存，从内存中返回数据的速度始终比去数据库查的速度快，因而可以大大提供应用程序的性能。毕竟现在内存非常便宜，用空间换取时间效率应该是非常划算的。尤其是对耗时比较长的、需要建立网络链接的数据库查询操作等。对于 Web Application 来讲，需要缓存的不仅仅是页面，很多时候还需要定制一些去缓存的内容，这时候就需要用到 HttpRun.Cache 对象。

（2）多用存储过程。

在 ASP.Net 项目中使用存储过程，首先可以提高数据库的安全性，其次可以提高运行 SQL 代码运行的速度，在大型项目中一般是必不可少的。

（3）数据库访问优化。

数据库访问性能优化是一项重要的工作，其中比较重要的是数据库的连接和关闭。在建立数据库连接后，只有在真正需要操作时才打开连接，使用完毕后马上关闭，从而缩短数据库连接打开的时间，避免出现超出连接限制的情况。

3. 配置优化

（1）多服务器部署，服务器的功能尽量单一，将 IO 操作尽量分布在不同的服务器上。例如，可以将大部分的 js 和 css 内容部署到一个静态服务器上，以减少主服务器的 I/O 操作。

（2）禁用调试模式。在部署生产应用程序或进行任何性能测量之前，始终记住禁用调试模式。如果启用了调试模式，应用程序的性能可能受到非常大的影响。

8.2.2 单元测试

单元测试对于一个软件或一个网站来说是至关重要的，对于软件的工作周期或是后期的扩展都是有很大影响的。

【实例 8.1】 有一个网站 MyAddWeb，实现两数相加，代码如下所示：

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.UI;
using System.Web.UI.WebControls;
using ch08AddDAL;
public partial class AddAB : System.Web.UI.Page
{  protected void Button1_Click(object sender, EventArgs e)
    {
        int num= new Class1 (). Add (Convert. ToInt32 (TextBox1. Text), Convert. ToInt32
        (TextBox2.Text));
        TextBox3.Text=num .ToString();
    }
}
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
namespace ch08AddDAL
{
    public class Class1
    {
        public int Add(int a, int b)
        {    return a+b;
        }
    }
}
```

【步骤 1】依次选择“工具” →“自定义”选项，弹出图 8.2 所示的对话框。



图 8.2 【自定义】对话框

【步骤 2】选择“命令”选项卡，在上下文菜单中选择“编辑器上下文菜单|代码窗口”，将“创建单元测试”通过“下移”按钮移到“运行测试”菜单下面，如图 8.3 和图 8.4 所示。

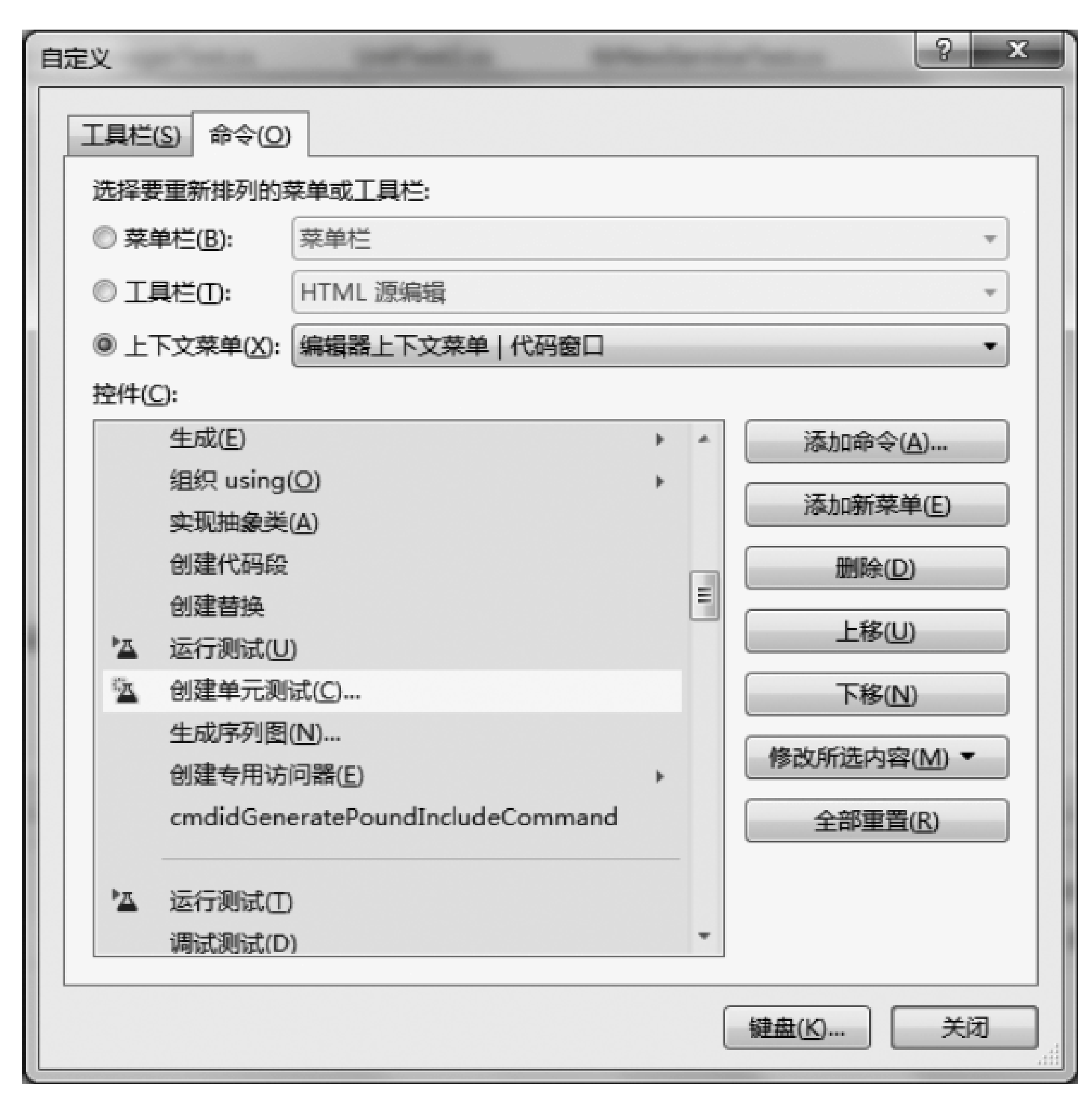


图 8.3 【命令】选项卡



图 8.4 添加“创建单元测试”

【步骤 3】重启后再对着类名，单击右键，发现上下文菜单中已经出现了“创建单元测试”按钮，但它是灰色的，并不能使用，如图 8.5 所示。



图 8.5 创建单元测试

【步骤 4】右击解决方案，单击“添加”→“新建项目”，弹出图 8.6 所示的对话框，左边选择“测试”，右边选择“单元测试项目”，输入名称，单击确定。新建的项目如图 8.7 所示。

【步骤 5】右击单元测试项目，依次选择“添加”→“单元测试”，如图 8.8 所示。

【步骤 6】对着类名，单击右键，就可以使用“创建单元测试”了，这时在“解决方案资源管理器”中可以看到添加了 Class1Test.cs 类，代码如下：

```
[TestClass()]
```

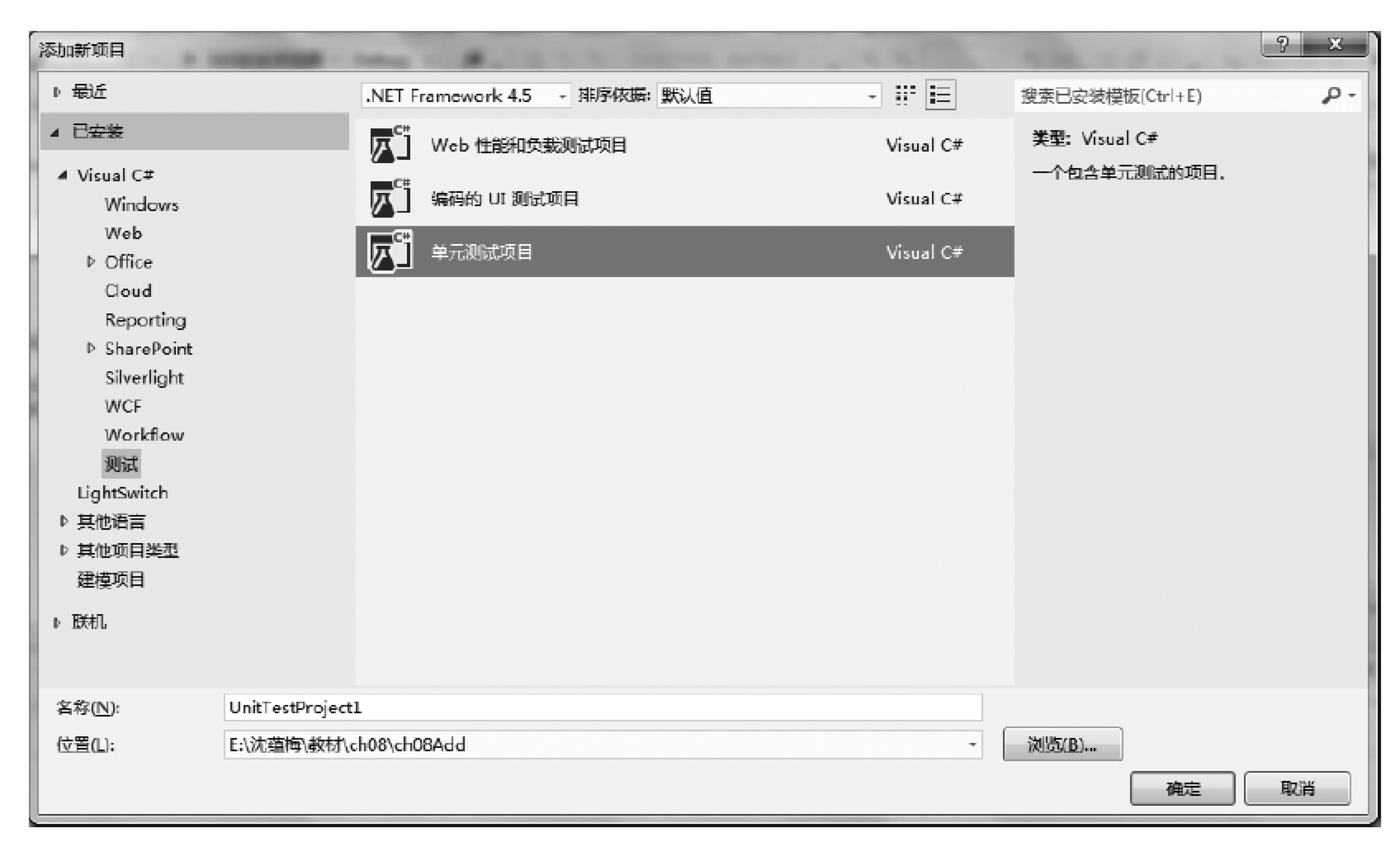


图 8.6 新建单元测试项目



图 8.7 解决方案资源管理器

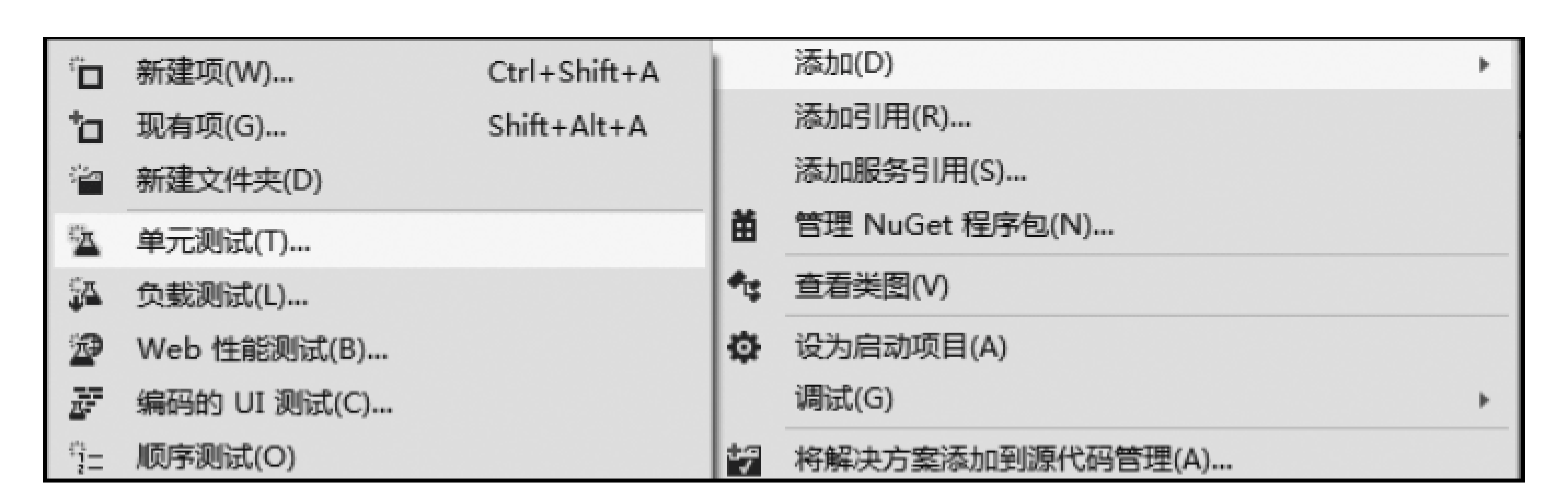


图 8.8 添加单元测试

```
    public class Class1Test
{
[TestMethod()]
        public void AddTest()
        {
            Class1 target=new Class1();        //TODO: 初始化为适当的值
            int a=0;                           //TODO: 初始化为适当的值
            int b=0;                           //TODO: 初始化为适当的值
            int expected=0;                    //TODO: 初始化为适当的值
            int actual;
```

```
            actual=target.Add(a, b);
            Assert.AreEqual(expected, actual);
            Assert.Inconclusive("验证此测试方法的正确性。");
        }
    }
```

说明：

1. [TestClass()]说明是一个单元测试类。
2. [TestMethod()]说明以下是一个测试用例。
3. int a=0；　　　　　　　　//TODO：初始化为适当的值
   int b=0；　　　　　　　　//TODO：初始化为适当的值

这两句是被测函数的输入参数，需要修改它的值，也就是我们输入测试用例的地方。

```
int expected=0;                //TODO: 初始化为适当的值
int actual;
actual=target.Add(a, b);
```

前一句是定义了期望值和对它进行初始化，后面是定义了实际值。

Assert 可以理解成断言：在 VSTS 里做单元测试是基于断言的测试。

8.3　项 目 训 练

通过对以上内容的学习，了解了系统优化的方法，同时了解了如何进行单元测试，现在我们回到项目导入的任务中来。

【步骤 1】搭建系统架构，如图 8.9 所示。

【步骤 2】新建数据库 DBYGCX，添加数据表，如图 8.10 所示。

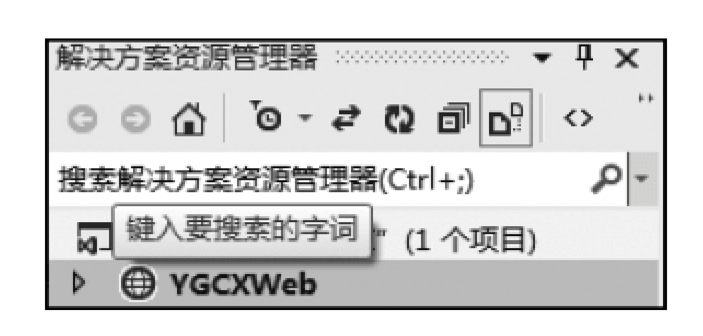


图 8.9　搭建系统框架

OEM-20130723TNF....CX - dbo.tbUserCX　SQLQuery1.sq

列名	数据类型	允许 Null 值
UserName	nvarchar(50)	☑
Sex	nchar(10)	☑
Depart	nvarchar(50)	☑
GongZi	int	☑
		☐

图 8.10　tbUserCX 数据表

【步骤 3】创建存储过程，代码如下：

```
create proc [dbo].[SelectParUsers]
(
  @name nvarchar(50)
)
as
begin
select *  from dbo.tbUserCX where Depart=@name
```

```
end
```

【步骤 4】配置 web. config。

```
<connectionStrings>
    <add name ="DefaultConnection" providerName ="System.Data.SqlClient"
    connectionString="Data Source=.;Initial Catalog=DBYGCX;Integrated Security=True" /
    >
  </connectionStrings>
```

【步骤 5】添加页面 YGCX. aspx,源视图代码如下:

```
<%@ Page Language="C#" AutoEventWireup="true" CodeFile="YGCX.aspx.cs" Inherits="YGCX"
%>
<!DOCTYPE html>
<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
<meta http-equiv="Content-Type" content="text/html; charset=utf-8"/>
    <title></title>
</head>
<body>
    <form id="form1" runat="server">
    <div>
    <table>
        <tr><td colspan="3" style="text-align:center">查询系统</td></tr>
        <tr><td>请输入部门名称</td><td>
            <asp:TextBox ID="txtDCX" runat="server"></asp:TextBox>
            </td><td>
                <asp:Button ID="btnCX" runat="server" Text="查询" OnClick="btnCX_
                Click" />
            </td></tr>
         <tr><td colspan="3" style="text-align:center">
             <asp:GridView ID="GridView1" runat="server" AutoGenerateColumns= "False">
                 <Columns>
                     <asp:BoundField DataField="UserName" HeaderText="姓名" />
                     <asp:BoundField DataField="Sex" HeaderText="性别" />
                     <asp:BoundField DataField="Depart" HeaderText="部门" />
                     <asp:BoundField DataField="GongZi" HeaderText="工资" />
                 </Columns>
             </asp:GridView>
             </td></tr>
    </table>
    </div>
    </form>
</body>
```

```
</html>
```

【步骤 6】后台编写代码如下：

```
protected void btnCX_Click(object sender, EventArgs e)
    {
        string con=System.Configuration.ConfigurationManager.
        ConnectionStrings["DefaultConnection"].ToString();
            SqlConnection conn=new SqlConnection(con);
            conn.Open();
          SqlCommand cmd=new SqlCommand("SelectParUsers", conn);
            cmd.CommandType=CommandType.StoredProcedure;
            SqlDataAdapter da=new SqlDataAdapter(cmd);
            cmd.Parameters.Add("@name", SqlDbType.NVarChar, 50).Value=txtDCX.Text.Trim
            ();
            DataTable ds=new DataTable();
            da.Fill(ds);
            GridView1.DataSource=ds;
            GridView1.DataBind();
            conn.Close();
    }
```

【步骤 7】运行结果如图 8.1 所示。

8.4 平行项目训练

1. 训练内容

实现用户名有效性验证，要求用 Ajax 来实现。

2. 训练目的

(1) 进一步训练和巩固学生对 Ajax 概念和原理的理解；

(2) 使学生对系统优化有一个比较深刻的印象和掌握。

3. 训练过程

【步骤 1】搭建系统框架并添加各层之间的依赖关系，如图 8.11 所示。

【步骤 2】新建数据库 dbUserAvail，并添加数据表，如图 8.12 所示，配置 Web.Config："<add name="DefaultConnection" providerName="System.Data.SqlClient" connectionString="Data Source=.;Initial Catalog=dbUserAvail;Integrated Security=True" />"。

【步骤 3】根据数据表添加相应类，如图 8.13 所示。

【步骤 4】实现服务器端的程序。



图 8.11 搭建系统框架

OEM-20130723TNF.d...vail - dbo.tbUsers

列名	数据类型	允许 Null 值
Name	nvarchar(50)	☑
Age	int	☑
Sex	nchar(10)	☑
JiGuan	nvarchar(50)	☑

图 8.12 tbUsers 数据表

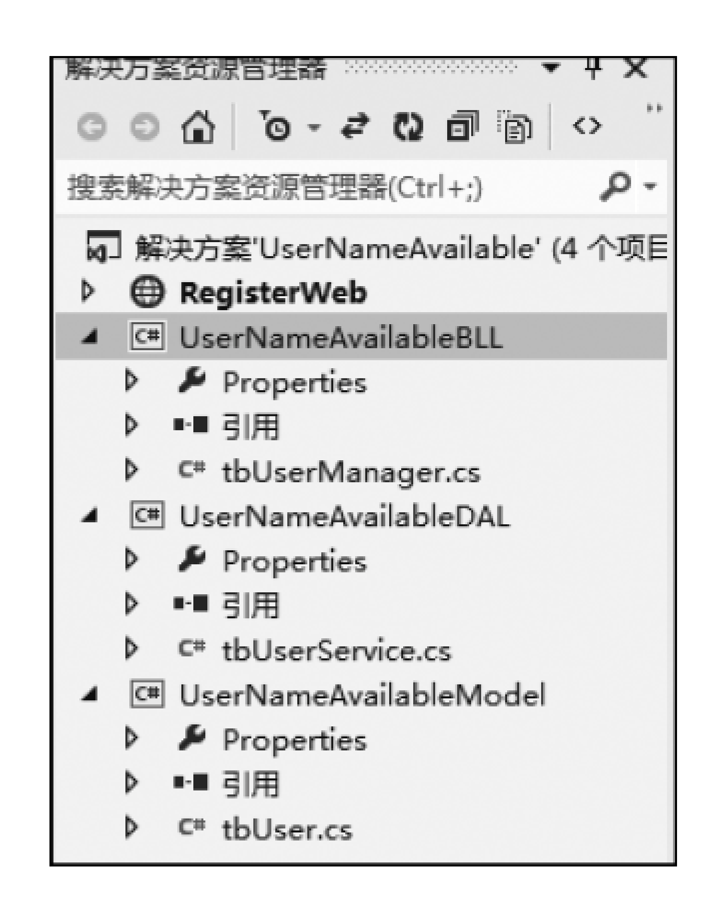


图 8.13 添加类

(1) 编写实体层代码。

```
public class tbUser
    {
        private string name;
        public string Name
        {
            get { return name; }
            set { name=value; }
        }
        private int age;
        public int Age
        {
            get { return age; }
            set { age=value; }
        }
        private string sex;
        public string Sex
        {
            get { return sex; }
            set { sex=value; }
        }
        private string jiGuan;
        public string JiGuan
        {
            get { return jiGuan; }
            set { jiGuan=value; }
        }
    }
```

（2）编写数据访问层代码。

```
using System;
using System.Collections.Generic;
using System.Configuration;
using System.Data.SqlClient;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace UserNameAvailableDAL
{
   public class tbUserService
    {
       public bool GetCountByName(string name)
       {
           string connString=ConfigurationManager.ConnectionStrings
           ["DefaultConnection"].ConnectionString;
           SqlConnection connection=new SqlConnection(connString);
           string sql=string.Format("select count(*)from dbo.tbUsers where Name='{0}'",
           name);
           SqlCommand command=new SqlCommand(sql, connection);
           connection.Open();
           int cout=(int)command.ExecuteScalar();
           connection.Close();
           return cout>0;
       }
    }
}
```

（3）编写业务逻辑层代码。

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using UserNameAvailableDAL;

namespace UserNameAvailableBLL
{
   public class tbUserManager
    {
        public bool GetCountByName(string name)
        {
```

```
                return new tbUserService().GetCountByName(name);
            }
        }
    }
```

（4）右击 RegisterWeb，依次选择“添加”→“添加新项”，弹出“添加新项”对话框，选择“C＃”和“一般处理程序”，在名称框中输入“TNameHandler. ashx”，如图 8.14 所示，单击“添加”按钮。



图 8.14　添加一般处理程序

（5）编写一般处理程序代码。

```
<%@ WebHandler Language="C#" Class="TNameHandler" %>
using System;
using System.Web;
using UserNameAvailableBLL;
public class TNameHandler : IHttpHandler {
        public void ProcessRequest(HttpContext context){
        context.Response.ContentType="text/plain";
        string name=context.Request.QueryString["UName"].Trim().ToString();
        if(new tbUserManager().GetCountByName(name))
        context.Response.Write("false");
        else
        context.Response.Write("true");        }
    public bool IsReusable {
        get {
            return false;
        }
    }
}
```

【步骤 5】实现 Ajax 客户端程序。

新建页面 Reg.aspx，编写代码如下：

```
<%@ Page Language="C#" AutoEventWireup="true" CodeFile="Reg.aspx.cs" Inherits="Reg" %>

<!DOCTYPE html>

<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
<meta http-equiv="Content-Type" content="text/html; charset=utf-8"/>
    <title></title>
    <script type="text/javascript">
        var xhr;
        function createXMLHttpRequest(){
            if(window.ActiveXObject){                    //如果是 IE 浏览器
                return new ActiveXObject("Microsoft.XMLHTTP");
            }
            else if(window.XMLHttpRequest){              //非 IE 浏览器
                return new XMLHttpRequest();
            }
        }
        function UserExit(uno){
            if(uno !=""){
                //请求字符串
                var url="TNameHandler.ashx?UName="+escape(uno);
                //1.创建 XMLHttpRequest 组件
                xhr=createXMLHttpRequest();
                //2.设置回调函数
                xhr.onreadystatechange=readyDo;
                //3.初始化 XMLHttpRequest 组件
                xhr.open("GET", url, true);
                //4.发送请求
                xhr.send(null);
            }
        }
        function readyDo(){
            if(xhr.readyState==4
             && xhr.status==200){
                var result=xhr.responseText;
                if(result=="true")
                    document.getElementById("mess").style.display="none";
                else
                    document.getElementById("mess").style.display="inline";
            }
```

```
            }
</script>
</head>
<body>
    <form id="form1" runat="server">
    <div>
    <table>
        <tr><td colspan="2" style="text-align:center">注册页面</td></tr>
        <tr><td>用户名</td><td>
            <asp:TextBox ID="txtName" runat="server" onblur="UserExit(this.value)"></
            asp:TextBox><span id="mess" style="display:none;color:red">该用户名已被
            注册过,请重新输入</span></td></tr>
        <tr><td>年龄</td><td>
            <asp:TextBox ID="txtAge" runat="server"></asp:TextBox></td></tr>
        <tr><td>性别</td><td>
            <asp:TextBox ID="txtSex" runat="server"></asp:TextBox></td></tr>
        <tr><td>籍贯</td><td>
            <asp:TextBox ID="txtJG" runat="server"></asp:TextBox></td></tr>
        <tr><td colspan="2" style="text-align:center">
            <asp:Button ID="Button1" runat="server" Text="注册" /></td></tr>
    </table>
    </div>
    </form>
</body>
</html>
```

【步骤 6】运行结果如图 8.15 所示。



图 8.15 运行结果

8.5 总 结

本章通过简单项目案例,介绍了如何优化与测试系统,介绍了系统优化的几种方法及用 VS 自带单元测试进行系统测试。通过项目训练“员工查询系统”及平行项目训练“用户名有效性验证”整体训练了创建存储过程和用 Ajax 来优化系统,提高系统整体性能。

8.6 习　　题

1. 简述如何进行单元测试。
2. 系统优化有哪几种方法？
3. 修改平行项目训练，用存储过程来验证用户名有效性，并实现注册功能。

第 9 章　系统发布与部署

本章要点：

- Web 服务器部署
- 域名使用
- 发布与浏览

技能目标：

- 会部署 Web 服务器
- 会使用域名
- 会发布与浏览网站

9.1　项目导入

【项目场景】

公司小刘自己开发了一个网站，想要申请一个域名，发布浏览网站，请你为小刘申请一个空间域名，并发布网站。

【问题引导】

(1) 如何配置 Web 服务器。
(2) 如何使用域名。
(3) 如何发布、浏览网站。

9.2　技术与知识准备

9.2.1　Web 服务器部署

Web 服务器又称为 WWW 服务器，它是放置一般网站的服务器。一台 Web 服务器上可以建立多个网站，各网站的拥有者只需要把做好的网页和相关文件放置在 Web 服务器的网站中，其他用户就可以用浏览器访问网站中的网页了。我们配置 Web 服务器，就是在服务器上建立网站，并设置好相关的参数。

9.2.2　域名使用

虽然可以通过 IP 地址来访问每一台主机，但是要记住那么多枯燥的数字串显然是非常困难的，为此，Internet 提供了域名(Domain Name)。域名也由若干部分组成，各部分之间用小数点分开，例如“苏州健雄职业技术学院”主机的域名是健雄的简拼，就是“wjxvtc.cn”，显然域名比 IP 地址好记忆多了。域名前加上传输协议信息及主机类型信息就构成了网址(URL)，例如“苏州健雄职业技术学院”的 www 主机的 URL 就是：“http://www.wjxvtc.cn ”。

人们习惯记忆域名，但机器间互相只认 IP 地址，域名与 IP 地址之间是一一对应的，它们之间的转换工作称为域名解析，域名解析需要由专门的域名解析服务器来完成，整个过程是自动进行的。

由于 Internet 最初是在美国发源的，因此最早的域名并无国家标识，人们按用途把它们分为几个大类，它们分别以不同的后缀结尾：

.com 用于商业公司；

.org 用于组织、协会等；

.net 用于网络服务；

.edu 用于教育机构；

.gov 用于政府部门；

.mil 用于军事领域。

随着 Internet 向全世界的发展，除了 edu、gov、mil 一般只在美国专用外，另外三个大类 com、org、net 则成为全世界通用，因此这三大类域名通常称为国际域名。由于国际域名资源有限，各个国家、地区在域名最后加上了国家标识段，由此形成了各个国家、地区自己的国内域名，如：

.com.cn 中国的商业；

.org.hk 香港的组织；

.net.jp 日本的网络。

显然国际域名具有比国内域名更高的级别，更有利于企业的形象。注册域名，为企业在 Internet 上建立信息宣传中心奠定了基础。绝大多数企业在建立自己的网页时希望使用企业的公司名或商标名作为域名。国际域名的资源十分有限，且不受商标法保护，谁先注册，谁就有权使用。

9.2.3　发布与浏览

视网站空间或服务器环境，有多种实施方案。如果网站是搭建在虚拟空间的，只需要在自己的域名解析管理中设置域名解析到空间 IP 即可；如果网站是搭建在本地自己的服务器上的，需要分析本地网络环境再决定实施方案。

因为本地网络环境可能比较复杂，如有可能是动态 IP，甚至无公网 IP，对应需要采用动态域名解析方案和 80 端口映射方案，来发布网站应用做网站服务。具体实施过程

如下：

1. 动态IP时，在本地内网安装并启用nat123动态域名解析。可以使用自己的顶级域名，固定在本地解析，如图9.1所示。



图9.1 动态IP域名解析

2. 无公网IP时，在内网启用nat123映射解析的80端口映射。配置80映射信息，外网地址是自己的域名。映射后，用域名即可以访问对应映射的网站应用，如图9.2所示。



图9.2 无公网IP域名解析

3. 使用自己的域名的同时，需要在自己的域名注册商网站的域名解析系统中，设置域名指向提示的DNS域名解析。

9.3 项目训练

通过对以上内容的学习，了解了Web服务器的部署、域名的使用以及发布网站的方法，现在我们回到项目导入的任务中来。

1. 配置IIS

一般在安装操作系统时不默认安装IIS，所以在第一次配置Web服务器时需要安装IIS。安装方法见第6章。

2. 申请并使用域名

【步骤1】注册账户。

动态域名有收费和免费之分，一般家庭都用免费的。在这里以 www.PubYun.com 为例。

(1) 打开 www.PubYun.com，单击右上角的注册按钮，如图 9.3 所示。



图 9.3　单击“注册”按钮

(2) 填写相关注册信息，如图 9.4 所示。

账户名：hellosym
账户名由3-30位字母a-z、数字0-9组成
邮箱地址：
需要用邮箱验证帐号，请输入常用的正确邮箱。
请再输一次邮箱：
为防止输错，请再输一次邮箱。
登录密码：●●●●●●●●●●
密码由6-30个字符组成。
确认密码：●●●●●●●●●●
手机号码：
按相关部门要求，PubYun进行实名制认证，必须输入正确的手机号码，接收验证码
验证码：K1H7　刷新
K1H7

图 9.4　填写注册信息

(3) 激活邮箱，如图 9.5 所示。

(4) 通过手机认证，如图 9.6 所示。

(5) 注册成功，如图 9.7 所示。

【步骤 2】申请免费域名。

(1) 登录 www.PubYun.com，输入用户名和密码，如图 9.8 所示。

(2) 找到 PubYunDNS，单击创建 DNS 解析。在下拉菜单中选择“动态域名设置”，如图 9.9 所示。



图 9.5 激活邮箱

第一步：填写账户信息
第二步：激活电子邮箱
第三步：实名制（手机
您的手机号码：
校验码已发送到手机
倒计时剩余秒数：47秒
验证码：
提交
<<如果验证失败，您还可以使用：电话验证

图 9.6 手机认证

领先的公有云提供商
QQ交谈
第二步：激活电子邮箱
账户激活成功！立即使用我们的服务。
恭喜您注册成功，您可以使用公云的服务。
进入管理

图 9.7 注册成功



图 9.8 登录页面



图 9.9　创建 DNS 解析

(3) 设置一个自己的域名，如 szjxshym，如图 9.10 所示。



图 9.10　设置自己的域名

(4) 单击查询按钮，只要这个域名还没有人用，就能注册成功，你的域名全称就是：szjxshym. f3322. net，如图 9.11 所示。



图 9.11　查询域名

（5）单击右下角的创建域名，弹出图9.12所示的对话框，提示申请域名成功。



图9.12　域名注册成功

【步骤3】使用域名。

（1）先登录网站www.pubyun.com，输入用户名及密码，单击“服务支持”→“下载客户端”，如图9.13所示。

（2）单击客户端V1.0，如图9.14所示。



图9.13　下载客户端



图9.14　选择适合的客户端

（3）按照向导安装客户端，如图9.15～图9.18所示。

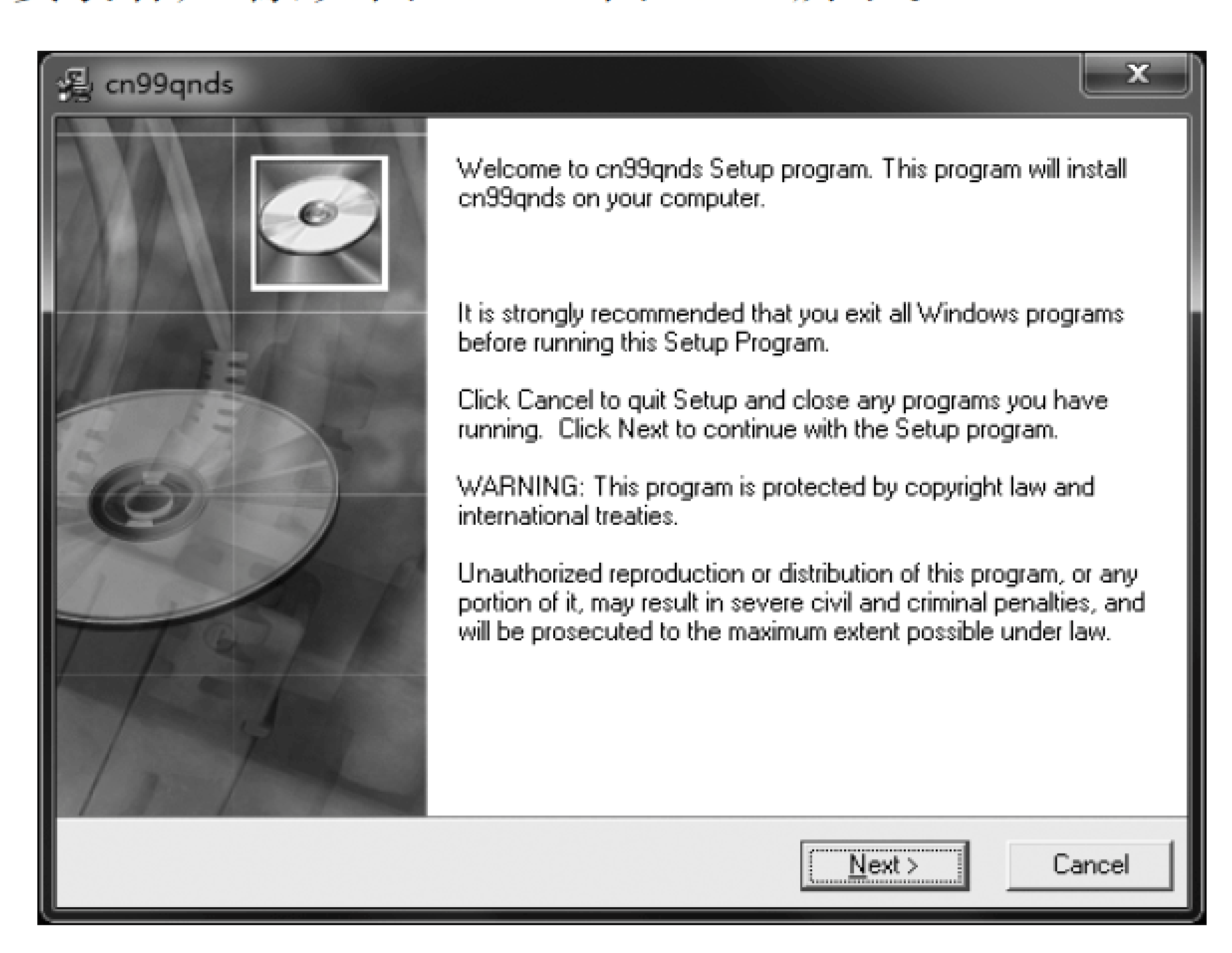


图9.15　安装客户端(一)



图 9.16　安装客户端(二)

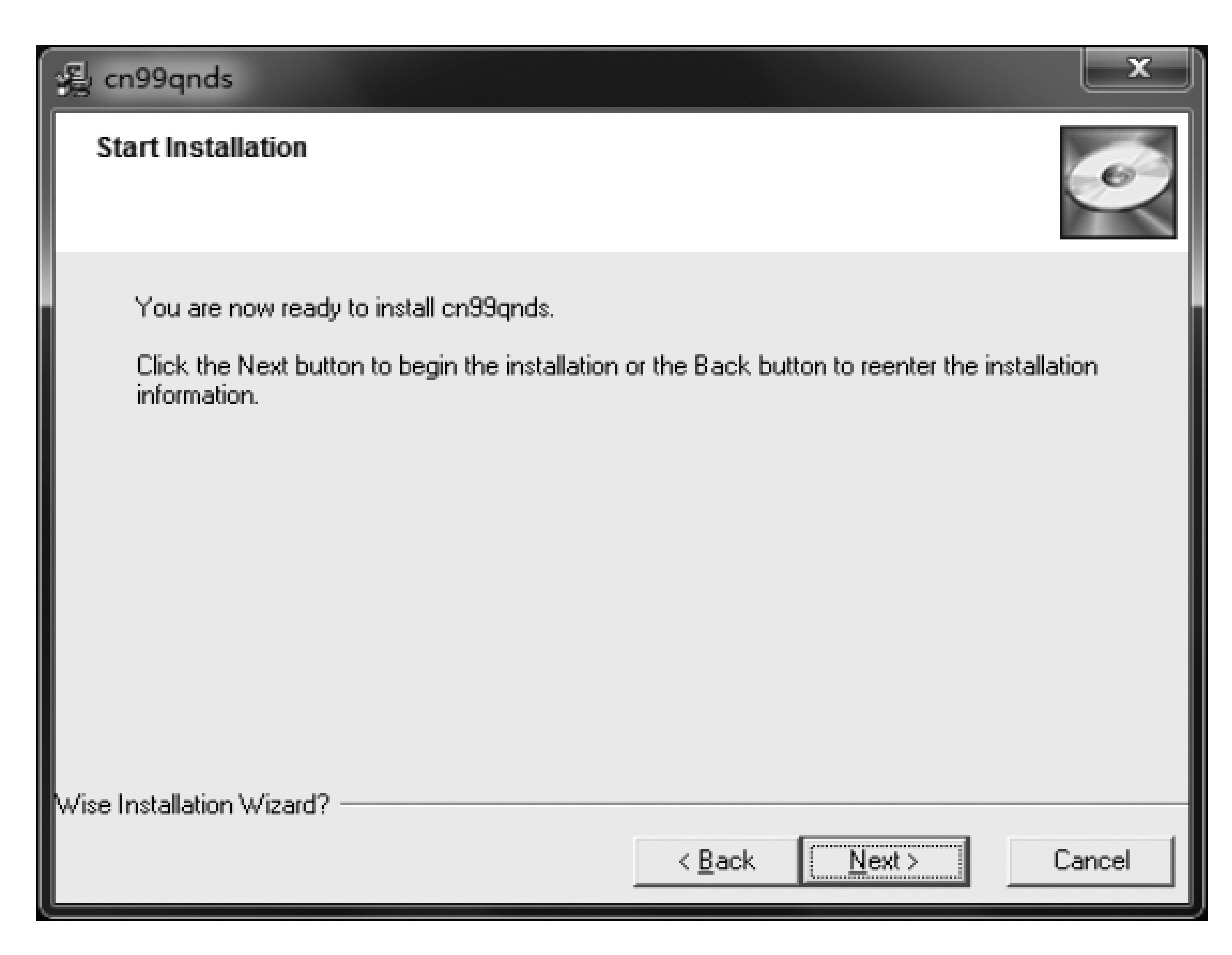


图 9.17　安装客户端(三)



图 9.18　安装客户端(四)

(4) 安装完成后,打开客户端,设置好登录的用户名、密码等,如图 9.19 所示。

图 9.19　设置用户名口令

(5) 成功后,屏幕右下角可以看到图标。

【步骤 4】发布、浏览网站。

(1) 打开 Internet 信息服务(IIS)管理器,如图 9.20 所示。

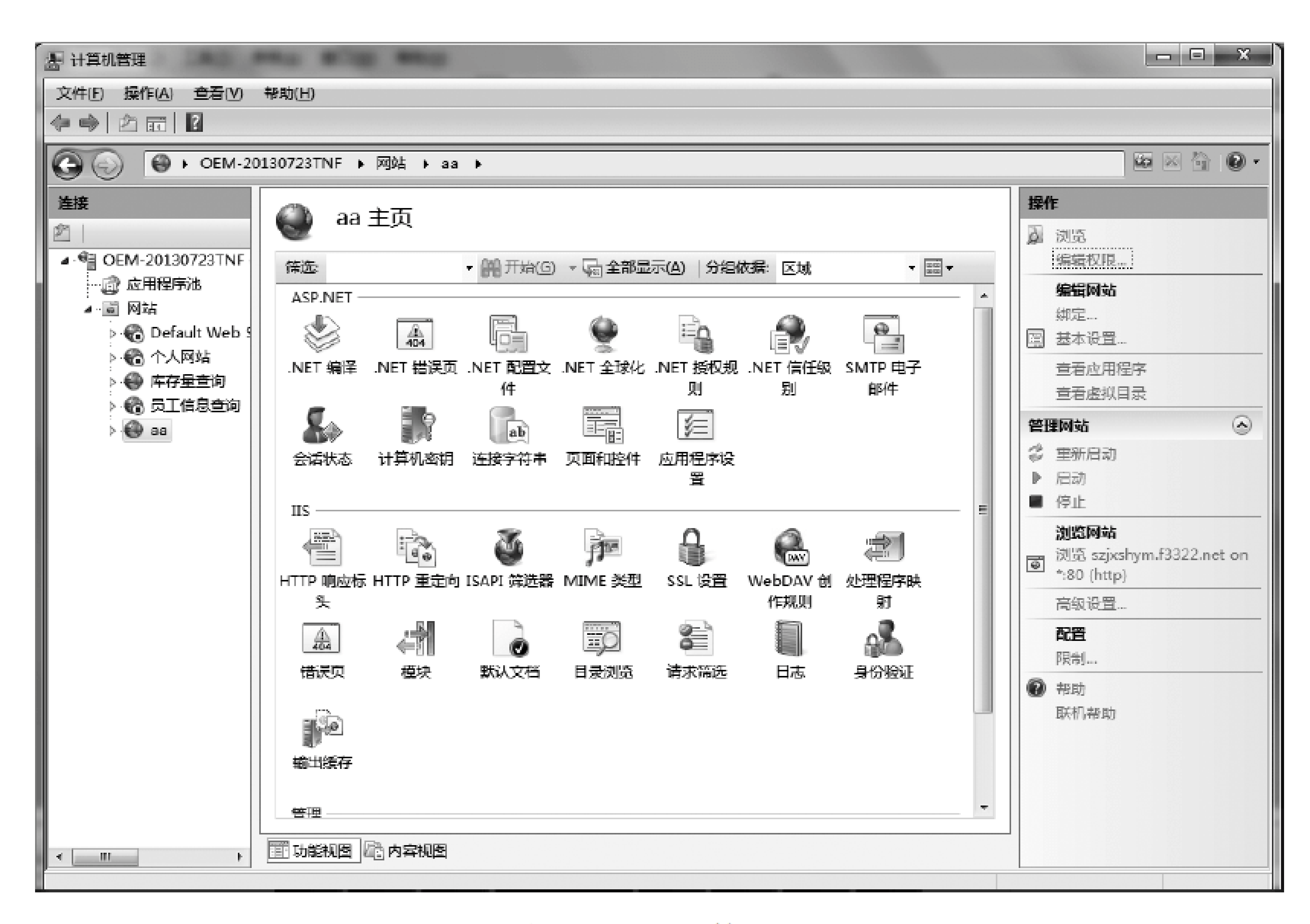


图 9.20　IIS 管理器

(2) 右击网站,选择添加网站选项,如图 9.21 所示。

(3) 弹出添加网站的窗口,在该窗口中,输入网站名称,物理路径(提前发布的网站文件夹),IP 地址选择全部未分配,主机名中输入申请的域名:szjxshym.f3322.net,如图 9.22 所示。

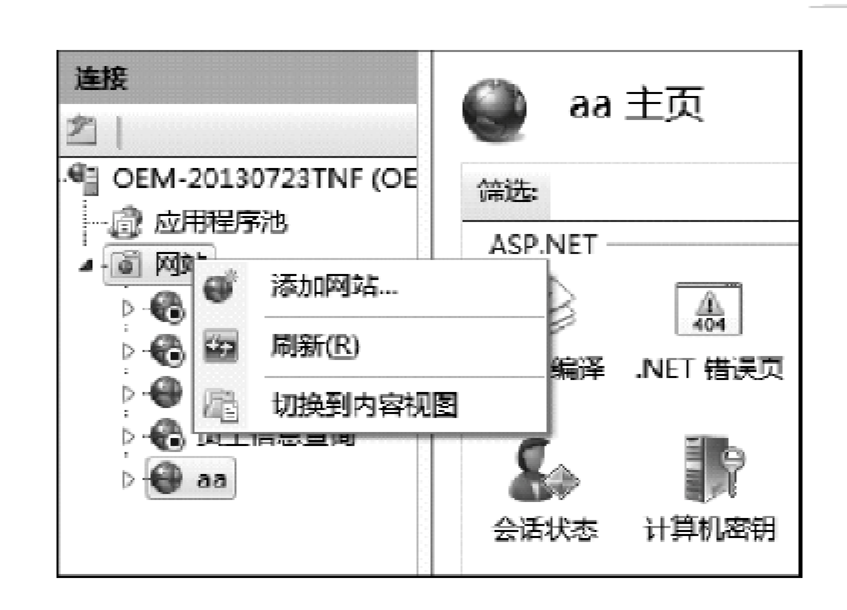


图 9.21　添加网站



图 9.22　添加网站对话框

（4）单击 Internet 信息服务(IIS)管理器窗口右侧的“浏览 szjxshym. f3322. net”，弹出图 9.23 所示的页面。



图 9.23　页面浏览

9.4 平行项目训练

1. 训练内容

Web 服务器部署、域名使用、发布与浏览网站。

2. 训练目的

进一步训练和巩固学生对 Web 服务器部署、域名使用、发布与浏览网站的能力。

3. 训练过程

【步骤 1】配置个人笔记本电脑 Web 服务器。

【步骤 2】申请免费域名，并使用。

【步骤 3】在个人笔记本电脑上发布学生制作完成的网站，并浏览该网站。

9.5 总结

本章介绍了 Web 服务器部署的原理及方法、域名的申请及使用方法，以及发布并浏览网站的方法，同时通过项目训练及平行项目整体训练了学生对 Web 服务器部署、域名申请及使用、发布与浏览网站的能力。

9.6 习题

1. 域名是什么？为什么要使用域名？与 IP 的关系是什么？
2. 简述域名的申请过程。
3. 自己申请域名，创建学生个人网站，发布网站。

第 10 章　综合项目实训

10.1　综合项目实训说明

该部分内容可以根据课程学时或学生的实际情况，有选择地开展。

10.1.1　实训目的

综合项目实训是完成课程教学计划的重要一环，有较强的实践性和综合性，对于帮助学生进一步理解课堂教学内容，培养学生的应用实践能力，为进一步学习更高阶段的课程打下基础。

综合项目实训是 ASP. NET 动态 Web 开发技术的配套训练，在课程教学最后阶段实施，实训目的是：

(1) 进一步巩固和加深学生对 ASP. NET 基础知识、技术、类的理解和掌握度，培养学生综合运用 ASP. NET 语言知识和技术分析解决实际问题的能力。

(2) 通过一个小型的信息管理系统，使学生了解项目开发过程，培养学生创造性思维，提高项目设计、编码与调试能力。

(3) 通过项目实训，使学生能够按照软件工程的基本方法开发小型的信息管理系统。

10.1.2　实训对象

面向计算机专业大二学生开设《ASP. NET 动态 Web 开发技术》，学生在掌握了一定的 C# 语法基础上，学习. NET 相关知识。

10.1.3　实训项目

选择“新闻发布系统”作为综合实训项目。

10.2　新闻发布系统

10.2.1　系统功能模块

在本系统中有两类用户：普通用户和管理员用户，两种不同的用户所具有的操作权

限和操作内容不同，普通用户可以实现新闻的浏览及查看每一条新闻详细信息，还能实现新闻的搜索功能。管理员在后台实现新闻的增删改查以及友情链接信息管理等。功能模块图如图10.1所示。

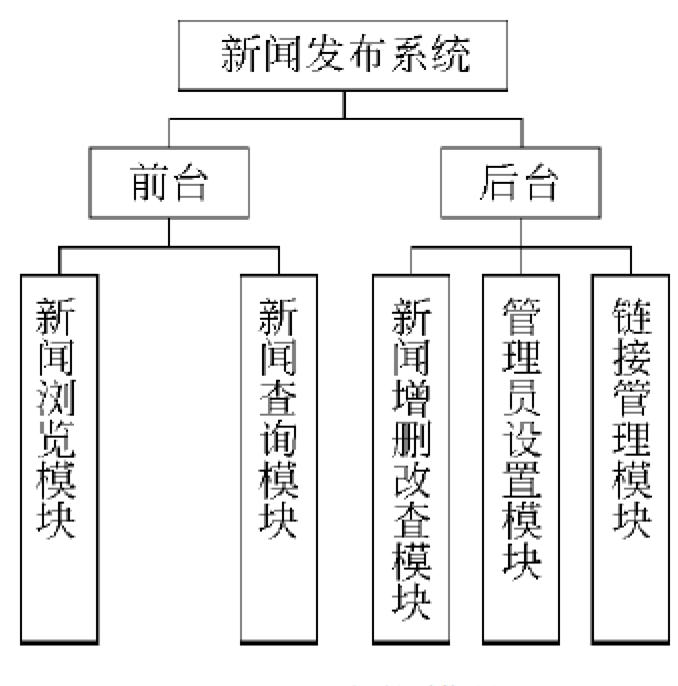


图10.1 功能模块图

10.2.2 数据库设计

数据库是信息管理系统的后台数据管理中心，一个信息管理系统的功能是否健全，关键在于对数据库的设计，只有对数据库进行合理的设计，才能开发出完善而有效的管理系统。

新闻发布系统的数据库中应该包括以下几张表格：新闻表(tbLink 表)、用户表(tbUser 表)、新闻类别表(tbStyle 表)和友情链接表(tbLink 表)。具体数据库创建及数据表新建见第1章内容。

10.2.3 系统详细设计与实现

10.2.3.1 新闻浏览模块的设计

1. 用户控件 menu.ascx

【步骤1】源视图代码如下：

```
<%@ Control Language="C#" AutoEventWireup="true" CodeFile="menu.ascx.cs" Inherits="
menu" %>
<link href="css/css.css" rel="stylesheet" type="text/css" />
<style type="text/css">
    .style1
    {
        height: 11px;
    }
</style>
<table class="css" style="width:801px;">
<tr>
```

```
<td  colspan="7" style="background-image:url(image/1.jpg); background-repeat:no-
repeat;height:127px">
</td>
</tr>
<tr>
 <td style="background-image: url(image/302.gif); background-repeat:no- repeat;width:
 114px; height: 44px; text-align:center">
 <asp:HyperLink ID="HyperLink1" runat="server" NavigateUrl="~/index.aspx" Font-
Underline="false">主页</asp:HyperLink>
 </td>
 <td style="background-image: url(image/302.gif); width: 114px; height: 44px; text-
align:center">
 <asp:HyperLink ID="HyperLink2" runat="server" NavigateUrl="~/newsList.aspx? id=1">通
 知公告</asp:HyperLink>
 </td>
 <td style="background-image: url(image/302.gif); width: 114px; height: 44px; text-
align:center">
 <asp:HyperLink ID="HyperLink3" runat="server" NavigateUrl="~/newsList.aspx? id=2">创
 业教育</asp:HyperLink>
 </td>
 <td style="background-image: url(image/302.gif); width: 114px; height: 44px; text-
align:center">
 <asp:HyperLink ID="HyperLink4" runat="server" NavigateUrl="~/newsList.aspx? id=3">教
 改动态</asp:HyperLink>
 </td>
 <td  style="background-image: url(image/302.gif); width: 114px; height: 44px; text-
align:center">
 <asp:HyperLink ID="HyperLink5" runat="server" NavigateUrl="~/newsList.aspx? id=4">校
 园新闻</asp:HyperLink>
 </td>
 <td style="background-image: url(image/302.gif); width: 114px; height: 44px; text-
align:center">
 <asp:HyperLink ID="HyperLink6" runat="server" NavigateUrl="~/newsList.aspx? id=5">学
 院海报</asp:HyperLink>
 </td>
 <td  style="background-image: url(image/302.gif); width: 114px; height: 44px; text-
align:center">
 <asp:HyperLink ID="HyperLink7" runat="server" NavigateUrl="~/newsList.aspx? id=6">学
 生工作</asp:HyperLink>
 </td>
 </tr>
 <tr>
 <td  style="text-align:right"  colspan="7" class="style1">
输入关键字:<asp:TextBox ID="TextBox1" runat="server"></asp:TextBox>
```

```
    <asp:DropDownList ID="DropDownList1" runat="server">
        <asp:ListItem>创业教育</asp:ListItem>
        <asp:ListItem>通知公告</asp:ListItem>
        <asp:ListItem>校园新闻</asp:ListItem>
        <asp:ListItem>教改动态</asp:ListItem>
        <asp:ListItem>学生工作</asp:ListItem>
        <asp:ListItem>学院海报</asp:ListItem>
    </asp:DropDownList>
    <asp:Button ID="btnSearch" runat="server" Text="站内搜索"
        onclick="btnSearch_Click" />
    设置主页 收藏本站
</td>
</tr></table>
```

效果如图 10.2 所示。



图 10.2　用户控件 menu.ascx

【步骤 2】数据访问层代码如下所示：

```
public List<tbNew> GetNews(string style, string content)
        {
            string sql=string.Format("select * from tbnews where StyleId='{0}' and
            content like '%{1}%'", GetIdByStyleName(style), content);
            return GetNewBySQL(sql);
        }
  public int GetIdByStyleName(string StyleName)
        {
            string sql="select Id from dbo.tbStyle where Name=@Name";
            int id=(int)DBHelper.ExecuteScalar(DBHelper.ConnectionString, CommandType.
            Text, sql, new SqlParameter("@Name", StyleName));
            return id;
        }
```

【步骤 3】业务逻辑层代码如下：

```
public List<tbNew> GetNews(string style, string content)
        {
            return new tbNewService().GetNews(style, content);
        }
```

【步骤 4】表示层代码如下：

```
protected void btnSearch_Click(object sender, EventArgs e)
    {
        Session["tool"]="都市新闻网络中心—>站内查询("+DropDownList1.Text+")
        ----输入关键字为"+TextBox1.Text+"";
        Session["search"]= new tbNewManager().GetNews(DropDownList1.Text, TextBox1.
        Text);
        Response.Redirect("search.aspx");
    }
```

2. 用户控件 left. ascx

【步骤 1】源视图代码如下：

```
<%@ Control Language="C#" AutoEventWireup="true" CodeFile="left.ascx.cs" Inherits="
left" %>
<link href="css/css.css" rel="stylesheet" type="text/css" />
<table style="width:210px" class="css">
<tr>
<td>
    <asp:Calendar ID="Calendar1" runat="server" CssClass="css">
     <OtherMonthDayStyle ForeColor="Blue" />
        <TitleStyle BackColor="#0B66B6" BorderColor="#0000CC" Font-Bold=
        "True" />
        <TodayDayStyle BackColor="LightSkyBlue" ForeColor="Black"  />
        <WeekendDayStyle BackColor="White" ForeColor="#FF3300" />
    </asp:Calendar>
</td>
</tr>
<tr>
<td style="background-image:url(image/8.gif);height: 62px">
</td>
</tr>
<tr>
<td>
    <asp:DataList  CssClass="css" ID="DataList1" runat="server">
    <ItemTemplate>
    <table>
    <tr>
    <td><a href="#">
    <%#StringUtility.CutString(Eval("Title"), 8)%></a>
    </td>
    </tr>
    </table>
    </ItemTemplate>
```

```
        </asp:DataList>
</td>
</tr>
<tr>
<td  style="background-image: url(image/友情链接.gif);height:49px">
</td>
</tr>
<tr>
<td>
        <asp:DataList ID="DataList2" runat="server"
            onitemcommand="DataList2_ItemCommand">
        <ItemTemplate>
        <table>
        <tr>
        <td style="text-align:center">
            <asp:Image ID="Image1" runat="server"  ImageUrl='<%#Eval("picPath")% >' Height="
            49px"/>
        </td>
        </tr>
        <tr>
        <td style="font-size:11pt;text-align:center">
            <asp: LinkButton  ID =" LinkButton1 "  runat =" server "  CommandName =" select "
            CommandArgument='<%#Eval("ID")%>'><%#Eval("linkName
        </td>
        </tr>
        </table>
        </ItemTemplate>
        </asp:DataList>
        </td>
</tr>
</table>
```

效果如图 10.3 所示。

图 10.3　用户控件 left. ascx

【步骤 2】数据访问层代码如下所示：

在 tbNewService. cs 类中

```
public List<tbNew>GetTenNews()
        {
            string sql="select top 10 * from tbNews
            order by IssueDate desc";
            return GetNewBySQL(sql);
        }
private static List<tbNew>GetNewBySQL(string sql)
        {
            List<tbNew>newlist=new List<tbNew>();
```

```
            SqlDataReader dr=DBHelper.ExecuteReader(DBHelper.
            ConnectionString, CommandType.Text, sql);
            while(dr.Read())
            {
                tbNew tbn=new tbNew();
                tbn.Content=Convert.ToString(dr["Content"]);
                tbn.ID=Convert.ToInt32(dr["ID"]);
                tbn.IssueDate=Convert.ToDateTime(dr["IssueDate"]);
                tbn.Title=Convert.ToString(dr["Title"]);
                tbn.Style = new tbStylService ().GetStyleById (Convert.ToInt32 (dr ["
                StyleId"]));
                tbn.Imageid=Convert.ToString(dr["imageid"]);
                newlist.Add(tbn);
            }
            dr.Close();
            return newlist;
        }
```

在 tbLinService.cs 类中

```
public List<tbLin> GetAllLinks()
        {
            string sql="select * from tbLink";
            List<tbLin> Linkslist=new List<tbLin>();
            SqlDataReader dr=DBHelper.ExecuteReader(DBHelper.
            ConnectionString, CommandType.Text, sql);
            while(dr.Read())
            {
                tbLin tbl=new tbLin();
                tbl.ID=Convert.ToInt32(dr["ID"]);
                tbl.picPath=Convert.ToString(dr["picPath"]);
                tbl.linkName=Convert.ToString(dr["linkName"]);
                tbl.linkAddress=Convert.ToString(dr["linkAddress"]);
                tbl.addDate=Convert.ToDateTime(dr["addDate"]);
                Linkslist.Add(tbl);
            }
            dr.Close();
            return Linkslist;
        }
public tbLin GetLinkById(int id)
        {
            string sql="select * from tbLink where ID=@ID";
            tbLin tbl=new tbLin();
            SqlDataReader dr=DBHelper.ExecuteReader(DBHelper.
            ConnectionString, CommandType.Text, sql, new SqlParameter("@ID", id));
```

```
            while(dr.Read())
            {
                tbl.ID=Convert.ToInt32(dr["ID"]);
                tbl.picPath=Convert.ToString(dr["picPath"]);
                tbl.linkName=Convert.ToString(dr["linkName"]);
                tbl.linkAddress=Convert.ToString(dr["linkAddress"]);
                tbl.addDate=Convert.ToDateTime(dr["addDate"]);
            }
            dr.Close();
            return tbl;
        }
```

【步骤 3】业务逻辑层代码如下：

在 tbNewManager. cs 类中

```
public List<tbNew> GetTenNews()
      {
          return new tbNewService().GetTenNews();
      }
```

在 tbLinManager. cs 类中

```
public tbLin GetLinkById(int id)
      {
          return new tbLinService().GetLinkById(id);
      }
      public List<tbLin> GetAllLinks()
      {
          return new tbLinService().GetAllLinks();
      }
```

【步骤 4】表示层代码如下：

```
protected void Page_Load(object sender, EventArgs e)
    {
        if(!IsPostBack)
        {
            DataList1.DataSource=new tbNewManager().GetTenNews();
            DataList1.DataBind();
            DataList2.DataSource=new tbLinManager().GetAllLinks();
            DataList2.DataBind();
        }
    }
   protected void DataList2_ItemCommand(object source, DataListCommandEventArgs e)
    {
        if(e.CommandName=="select")
        {
```

```
            int id=Convert.ToInt32(e.CommandArgument);
            tbLin tblin=new tbLin();
            tblin=new tbLinManager().GetLinkById(id);
            string strLink=tblin.linkAddress;
            Response.Write("<script language=javascript>window.open('http://"+strLink
            +"')</script>");
        }
    }
```

3. 用户控件 bottom.ascx

源视图代码如下：

```
<%@ Control Language="C#" AutoEventWireup="true" CodeFile="bottom.ascx.cs" Inherits="
bottom" %>
<table style="width: 803px; height: 59px; background-color: #ffffff;text-align:center"
class="txt" border="0">
            <tr>
                <td  class="txt" colspan="3" style="height: 20px;text-align:center">
                    健雄职业技术学院 客户服务热线::(0512)53948888      
                    客户服务邮箱:mingrisoft@mingrisoft.com</td>
            </tr>
            <tr>
                <td class="css" style="height: 16px">
                </td>
                <td class="txt" style="width: 555px; height: 16px;text-align:center">
                    CopyRight <span lang="EN-US" style="font-size: 10.5pt; font-family: '
                    Times New Roman'">
                        ?  健雄职业技术学院 网络开发  
                        <asp:HyperLink ID="HyperLink1" runat="server" CssClass="txt"
                        NavigateUrl =" HouAdmin/HouLogin. aspx " > 后 台 入 口
                        </asp:HyperLink></span></td>
                <td class="css" style="height: 16px">
                </td>
            </tr>
        </table>
```

效果如图 10.4 所示。

苏州健雄职业技术学院 客户服务热线：(0512)53948888　客户服务邮箱：mingrisoft@mingrisoft.com
CopyRight ? 苏州健雄职业技术学院 网络开发　后台入口

图 10.4　用户控件 bottom.ascx

4. index.aspx 主界面

【步骤 1】源视图代码如下：

```
<%@ Page Language="C#" AutoEventWireup="true" CodeFile="index.aspx.cs"
Inherits="index" %>
<%@ Register src="left.ascx" tagname="left" tagprefix="uc1" %>
<%@ Register src="menu.ascx" tagname="menu" tagprefix="uc2" %>
<%@ Register src="bottom.ascx" tagname="bottom" tagprefix="uc3" %>
<!DOCTYPE html>
<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
<meta http-equiv="Content-Type" content="text/html; charset=utf-8"/>
    <title></title>
</head>
<body>
    <form id="form1" runat="server">
    <div style=" text-align:center">
    <table class="txt">
    <tr>
    <td colspan="2">
        <uc2:menu ID="menu1" runat="server" />
    </td>
    </tr>
     <tr>
    <td>
        <uc1:left ID="left1" runat="server" />
    </td>
    <td   style="width:590px;vertical-align:top">
    <table style="width:590px">
    <tr>
    <td class="style2">通知公告</td>
     <td style=" text-align:right" class="style2">更多>>><a href="newsList.aspx?id=
     1"><asp:Image ID="Image5" runat="server" ImageUrl="~/image/14.gif" /></a>
    </td>
    </tr>
    <tr>
    <td colspan="2">
    <table><tr>
    <td>
        <asp:Image ID="Image1" runat="server" ImageUrl="~/image/szyw.jpg" />
    </td>
    <td>
        <asp:DataList ID="DataList1" runat="server">
        <ItemTemplate>
        <table>
        <tr>
        <td>
```

```
        <a href="showNews.aspx?id=<%#Eval("ID")%>">
        <%#Eval("Title")%></a>
        </td>
        </tr>
        </table>
        </ItemTemplate>
        </asp:DataList>
        </td>
    </tr>
        </table>
    </td>
    </tr>
    <tr>
    <td>创业教育</td>
     <td style=" text-align:right">更多>>><a href="newsList.aspx?id=2"><asp:Image
     ID="Image6" runat="server" ImageUrl="~/image/14.gif" /></a>
    </td>
    </tr>
  <tr>
    <td colspan="2">
    <table><tr>
    <td>
        <asp:Image ID="Image2" runat="server" ImageUrl="~/image/jjdx.jpg" />
    </td>
    <td>
        <asp:DataList ID="DataList2" runat="server">
        <ItemTemplate>
        <a href="showNews.aspx?id=<%#Eval("ID")%>">
        <%#Eval("Title")%></a>
        </ItemTemplate>
        </asp:DataList>
        </td>
    </tr>
        </table>
    </td>
    </tr>
  <tr>
    <td>教改动态
    </td>
     <td style=" text-align:right">更多>>><a href="newsList.aspx?id=3"><asp:Image
     ID="Image7" runat="server" ImageUrl="~/image/14.gif" /></a>
    </td>
    </tr>
  <tr>
```

```
  <td colspan="2">
  <table><tr>
  <td>
      <asp:Image ID="Image3" runat="server" ImageUrl="~/image/sjjs.jpg" />
  </td>
  <td>
      <asp:DataList ID="DataList3" runat="server">
      <ItemTemplate>
      <a href="showNews.aspx?id=<%#Eval("ID")%>">
      <%#Eval("Title")%></a>
      </ItemTemplate>
      </asp:DataList>
      </td>
  </tr>
      </table>
  </td>
  </tr>
  <tr>
  <td>校园新闻</td>
   <td style=" text-align:right">更多>>><a href="newsList.aspx?id=4"><asp:Image
   ID="Image8" runat="server" ImageUrl="~/image/14.gif" /></a>
  </td>
  </tr>
<tr>
  <td colspan="2">
  <table><tr>
  <td>
      <asp:Image ID="Image4" runat="server" ImageUrl="~/image/kxjy.gif" />
  </td>
  <td>
      <asp:DataList ID="DataList4" runat="server">
      <ItemTemplate>
      <a href="showNews.aspx?id=<%#Eval("ID")%>">
      <%#Eval("Title")%></a>
      </ItemTemplate>
      </asp:DataList>
      </td>
  </tr>
      </table>
  </td>
  </tr>
   <tr>
  <td>学院海报</td>
   <td style=" text-align:right">更多>>><a href="newsList.aspx?id=5"><asp:Image
```

```
 ID="Image9" runat="server" ImageUrl="~/image/14.gif" /></a>
  </td>
  </tr>
<tr>
  <td colspan="2">
  <table><tr>
  <td>
      <asp:Image ID="Image10" runat="server" ImageUrl="~/image/fzdd.jpg" />
  </td>
  <td>
      <asp:DataList ID="DataList5" runat="server">
      <ItemTemplate>
      <a href="showNews.aspx?id=<%#Eval("ID")%>">
      <%#Eval("Title")%></a>
      </ItemTemplate>
      </asp:DataList>
      </td>
  </tr>
      </table>
  </td>
  </tr>
   <tr>
  <td>学生工作</td>
   <td style=" text-align:right">更多>>><a href="newsList.aspx?id=6"><asp:Image
 ID="Image11" runat="server" ImageUrl="~/image/14.gif" /></a>
  </td>
  </tr>
<tr>
  <td colspan="2">
  <table><tr>
  <td>
      <asp:Image ID="Image12" runat="server" ImageUrl="~/image/shxx.jpg" />
  </td>
  <td>
      <asp:DataList ID="DataList6" runat="server">
      <ItemTemplate>
      <a href="showNews.aspx?id=<%#Eval("ID")%>">
      <%#Eval("Title")%></a>
      </ItemTemplate>
      </asp:DataList>
      </td>
  </tr>
      </table>
  </td>
```

```
        </tr>
        </table>
        </td>
        </tr>
            <tr>
        <td colspan="2">
            <uc3:bottom ID="bottom1" runat="server" />
        </td>
                </tr>
        </table>
        </div>
        </form>
</body>
</html>
```

【步骤 2】数据访问层代码如下：

```
public List<tbNew> GetPartNews(string style)
        {
            string sql="select top 5 *  from tbNews where StyleId=@StyleId  order by
            IssueDate desc";
            List<tbNew> newlist=new List<tbNew>();
            SqlDataReader dr=DBHelper.ExecuteReader(DBHelper.
            ConnectionString, CommandType.Text, sql, new SqlParameter
            ("@StyleId", GetIdByStyleName(style)));
            while(dr.Read())
            {
                tbNew tbn=new tbNew();
                tbn.Content=Convert.ToString(dr["Content"]);
                tbn.ID=Convert.ToInt32(dr["ID"]);
                tbn.IssueDate=Convert.ToDateTime(dr["IssueDate"]);
                tbn.Title=Convert.ToString(dr["Title"]);
                tbn.Style = new tbStylService().GetStyleById(Convert.ToInt32(dr["
                StyleId"]));
                tbn.Imageid=Convert.ToString(dr["imageid"]);
                newlist.Add(tbn);
            }
            dr.Close();
            return newlist;
        }
```

【步骤 3】业务逻辑层代码如下：

```
public List<tbNew> GetPartNews(string style)
        {
            return new tbNewService().GetPartNews(style);
```

```
        }
```

【步骤 4】表示层代码如下：

```
protected void Page_Load(object sender, EventArgs e)
    {
        if(!IsPostBack)
        {
            DataList1.DataSource=new tbNewManager().GetPartNews("通知公告");
            DataList1.DataBind();
            DataList2.DataSource=new tbNewManager().GetPartNews("创业教育");
            DataList2.DataBind();
            DataList3.DataSource=new tbNewManager().GetPartNews("教改动态");
            DataList3.DataBind();
            DataList4.DataSource=new tbNewManager().GetPartNews("校园新闻");
            DataList4.DataBind();
            DataList5.DataSource=new tbNewManager().GetPartNews("学院海报");
            DataList5.DataBind();
            DataList6.DataSource=new tbNewManager().GetPartNews("学生工作");
            DataList6.DataBind();
        }
    }
```

【步骤 5】页面浏览如图 10.5 所示。



图 10.5　index.aspx 页面

10.2.3.2 新闻增删改查模块的设计

1. 母版页 HouAdmin.master

源视图代码如下：

```
<div style="text-align:center">
        <table style="width: 800px; height: 392px;background-color: #0b66b6;text-align:
        center"  class="txt">
            <tr>
                <td colspan="2" style="background-image: url(../image/后台.jpg);
                width: 211px;height: 142px;">

                </td>
            </tr>
            <tr>
                <td style="height: 210px; width: 211px; background-image: url(image/都
                市网络新闻中心后台页_12.gif); text-align:center; vertical-align:top"
                >
                    <asp:TreeView ID="TreeView1" runat="server">
                        <Nodes>
                            <asp:TreeNode Text="显示所有新闻" Value="显示所有新闻"
                            NavigateUrl="Default.aspx"></asp:TreeNode>
                            <asp:TreeNode Text="后台安全退出" Value="安全退出"></
                            asp:TreeNode>
                            <asp:TreeNode Text="通知公告" Value="通知公告">
                                <asp:TreeNode Text="添加通知公告" Value="添加通知公
                                告" NavigateUrl="NewAdd.aspx?id=1"></asp:TreeNode>
                                <asp:TreeNode Text="编辑通知公告" Value="编辑通知公
                                告" NavigateUrl="list.aspx?id=1"></asp:TreeNode>
                            </asp:TreeNode>
                            <asp:TreeNode Text="创业教育" Value="创业教育">
                                <asp:TreeNode Text="添加创业教育" Value="添加创业教
                                育" NavigateUrl="NewAdd.aspx?id=2"></asp:TreeNode>
                                <asp:TreeNode Text="编辑创业教育" Value="编辑创业教
                                育" NavigateUrl="list.aspx?id=2"></asp:TreeNode>
                            </asp:TreeNode>
                            <asp:TreeNode Text="教改动态" Value="教改动态">
                                <asp:TreeNode Text="添加教改动态" Value="添加教改动
                                态" NavigateUrl="NewAdd.aspx?id=3"></asp:TreeNode>
                                <asp:TreeNode Text="编辑教改动态" Value="编辑教改动
                                态" NavigateUrl="list.aspx?id=3"></asp:TreeNode>
                            </asp:TreeNode>
                            <asp:TreeNode Text="校园新闻" Value="校园新闻">
                                <asp:TreeNode Text="添加校园新闻" Value="添加校园新
```

```
                闻" NavigateUrl="NewAdd.aspx?id=4"></asp:TreeNode>
                <asp:TreeNode Text="编辑校园新闻" Value="编辑校园新
                闻" NavigateUrl="list.aspx?id=4"></asp:TreeNode>
            </asp:TreeNode>
            <asp:TreeNode Text="学院海报" Value="学院海报">
                <asp:TreeNode Text="添加学院海报" Value="添加学院海
                报" NavigateUrl="NewAdd.aspx?id=5"></asp:TreeNode>
                <asp:TreeNode Text="编辑学院海报" Value="编辑学院海
                报" NavigateUrl="list.aspx?id=5"></asp:TreeNode>
            </asp:TreeNode>
            <asp:TreeNode Text="学生工作" Value="学生工作">
                <asp:TreeNode Text="添加学生工作" Value="添加学生工
                作" NavigateUrl="NewAdd.aspx?id=6"></asp:TreeNode>
                <asp:TreeNode Text="编辑学生工作" Value="编辑学生工
                作" NavigateUrl="list.aspx?id=6"></asp:TreeNode>
            </asp:TreeNode>
            <asp:TreeNode Text="管理员设置" Value="管理员设置">
                <asp:TreeNode Text="添加管理员" Value="添加管理员"
                NavigateUrl="userAdd.aspx"></asp:TreeNode>
                <asp:TreeNode Text="编辑管理员" Value="编辑管理员"
                NavigateUrl="userEdit.aspx"></asp:TreeNode>
            </asp:TreeNode>
            <asp:TreeNode Text="链接管理" Value="链接管理">
                <asp:TreeNode Text="添加链接信息" Value="添加链接信
                息" NavigateUrl="linkAdd.aspx"></asp:TreeNode>
                <asp:TreeNode Text="编辑链接信息" Value="编辑链接信
                息" NavigateUrl="linkEdit.aspx"></asp:TreeNode>
            </asp:TreeNode>
        </Nodes>
    </asp:TreeView>
</td>
<td  style="height: 21px; background-image: url(image/都市网络新闻中
心后台页_13.gif);text-align:left; vertical-align:top">
 <asp:ContentPlaceHolder id="ContentPlaceHolder1" runat= "server">
        </asp:ContentPlaceHolder>
</td>
</tr>
</table>
<table style="width: 800px; height: 59px; background-color: #0b66b6;text-align:
center">
    <tr>
        <td class="txt" colspan="3" style="height: 20px;text-align:center">
          健雄职业技术学院 客户服务热线::(0512)53948888       客
          户服务邮箱:mingrisoft@mingrisoft.com</td>
```

```
            </tr>
            <tr>
                <td class="css" style="height: 16px">
                </td>
                <td  class="txt" style="width: 555px; height: 16px; text-align:center"
                >
                    CopyRight <span lang="EN-US" style="font-size: 10.5pt">
                        ©  健雄职业技术学院 网络开发    </span></td>
                <td class="css" style="height: 16px">
                </td>
            </tr>
        </table>
    </div>
```

效果如图 10.6 所示。

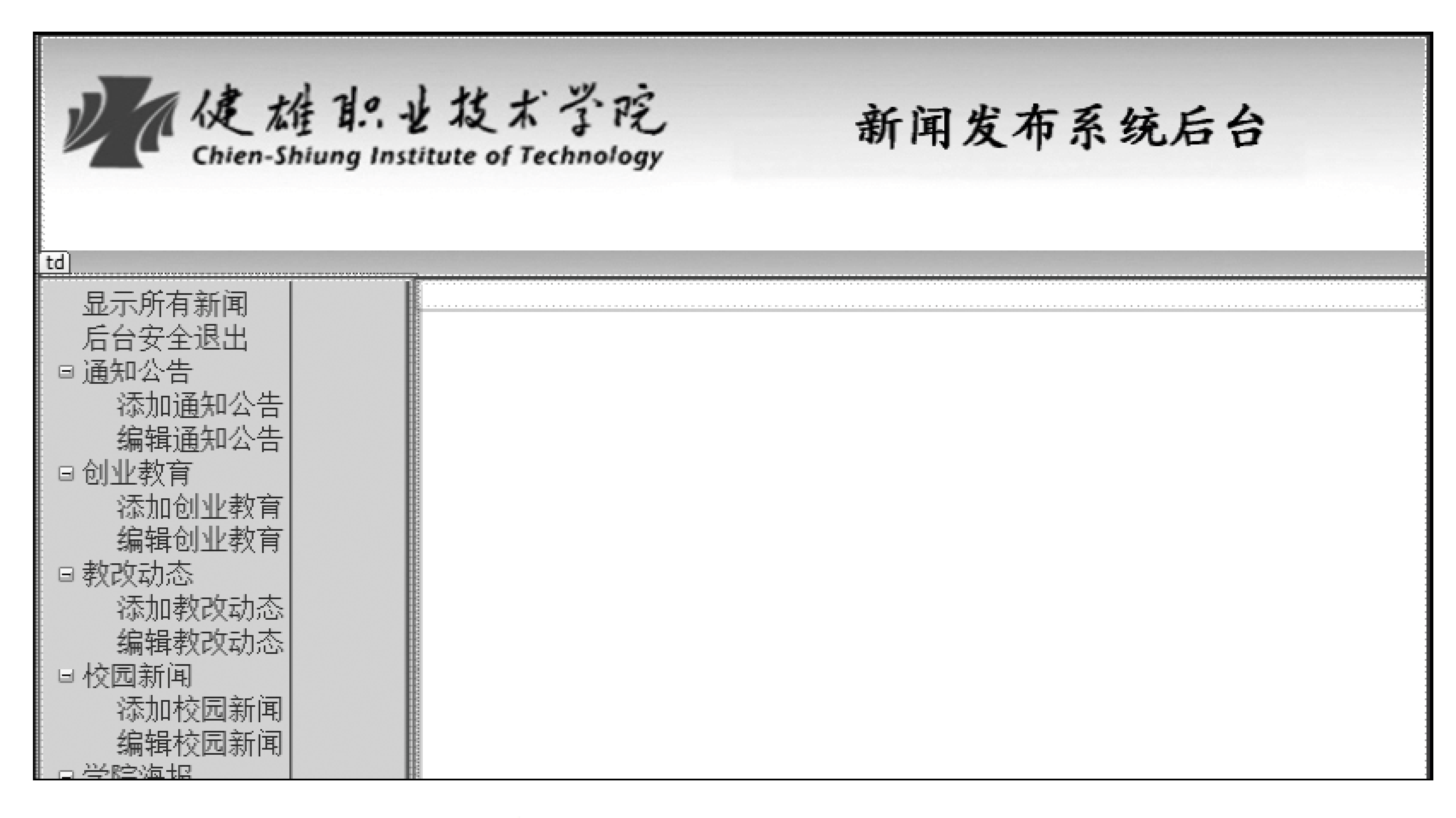


图 10.6 HouAdmin.master

2. 新闻编辑页面 list.aspx

【步骤 1】源视图代码如下：

```
<table style="width:580px">
<tr>
<td style="width:570px;text-align:center">
<asp:GridView ID="GridView1" runat="server" AllowPaging="True"
        AutoGenerateColumns="False" PageSize="30"
        onrowediting="GridView1_RowEditing" onrowdeleting="GridView1_RowDeleting"
        onrowupdating="GridView1_RowUpdating"
        onrowcancelingedit="GridView1_RowCancelingEdit"
        onrowdatabound="GridView1_RowDataBound" >
        <Columns>
        <asp:TemplateField HeaderText="ID">
```

```
    <ItemTemplate>
     <asp:Label ID="lblID" runat="server" Text='<%#Bind("ID")%>' Width= "30px"></
     asp:Label>
     </ItemTemplate>
    </asp:TemplateField>
        <asp:TemplateField HeaderText="新闻标题">
            <EditItemTemplate>
                <asp:TextBox ID="txtTitle" runat="server" Text='<%#Bind
                ("Title")%>' Width="190px" ></asp:TextBox>
            </EditItemTemplate>
             <ItemTemplate>
                <asp:Label ID="lblTitle" runat="server" Text='<%#Bind ("Title")%
              >' Width="190px"></asp:Label>
            </ItemTemplate>
        </asp:TemplateField>
        <asp:TemplateField HeaderText="发布日期">
            <ItemTemplate>
                <asp:Label ID="lblIssueDate" runat="server" Text='<%#Bind("
                IssueDate", "{0:d}")%>'
                    Width="110px"></asp:Label>
            </ItemTemplate>
            <EditItemTemplate>
                <asp:TextBox ID="txtIssueDate" runat="server" Text='<%#Bind("
                IssueDate", "{0:d}")%>'
                    Width="110px"></asp:TextBox>
            </EditItemTemplate>
        </asp:TemplateField>
        <asp:CommandField ShowEditButton="True"  />
        <asp:TemplateField HeaderText="自定义删除">
            <ItemTemplate>
                <asp:LinkButton ID="LinkButton1" runat="server" Width= "40px"
                CommandName="Delete"  CommandArgument='<%#Eval ("ID")%>'>删除</
                asp:LinkButton>
            </ItemTemplate>
        </asp:TemplateField>
    </Columns>
  </asp:GridView>
</td>
</tr>
</table>
```

【步骤 2】数据访问层代码如下：

```
public List<tbNew>GetPartNewsByStyle1(string style)
        {
```

```
        string sql;
        if(style=="")
            sql="select * from tbNews";
        else
         sql= string. Format ( " select  *  from tbNews where StyleId = ' {0} ' ",
         GetIdByStyleName(style));
        List<tbNew>newlist=new List<tbNew>();
        SqlDataReader dr=DBHelper.ExecuteReader(DBHelper.
        ConnectionString, CommandType.Text, sql);
        while(dr.Read())
        {
            tbNew tbn=new tbNew();
            tbn.Content=Convert.ToString(dr["Content"]);
            tbn.ID=Convert.ToInt32(dr["ID"]);
            tbn.IssueDate=Convert.ToDateTime(dr["IssueDate"]);
            tbn.Title=Convert.ToString(dr["Title"]);
            tbn.Imageid=Convert.ToString(dr["imageid"]);
            newlist.Add(tbn);
        }
        dr.Close();
        return newlist;
    }
    public bool ModifyNew(tbNew tbn)
    {
        string sql="update tbNews set Title=@Title,IssueDate=@IssueDate where ID=
       @ID";
        SqlParameter[] para=new SqlParameter[]
       {
           new SqlParameter("@Title",tbn.Title),
           new SqlParameter("@IssueDate",tbn.IssueDate),
           new SqlParameter("@ID",tbn.ID),
       };
        int n=DBHelper.ExecuteNonQuery(DBHelper.ConnectionString,
        CommandType.Text, sql, para)>0;
    }
    public bool DeteteNewById(int id)
    {
        string sql="delete from dbo.tbNews where ID=@ID";
        int n = DBHelper. ExecuteNonQuery (DBHelper. ConnectionString, CommandType.
        Text, sql, new SqlParameter("@ID", id))>0;
    }
```

【步骤 3】业务逻辑层代码如下：

```
public List<tbNew>GetPartNewsByStyle1(string style)
```

```
        {
            return new tbNewService().GetPartNewsByStyle1(style);
        }
        public bool ModifyNew(tbNew tbn)
        {
            return new tbNewService().ModifyNew(tbn);
        }
        public bool DeteteNewById(int id)
        {
            return new tbNewService().DeteteNewById(id);
        }
```

【步骤 4】表示层代码如下：

```
static string style="";
    protected void Page_Load(object sender, EventArgs e)
    {
        if(!IsPostBack)
        {
            int n=Convert.ToInt32(Request.QueryString["id"]);
            switch(n)
            {
                case 1:
                    style="通知公告";
                    break;
                case 2:
                    style="创业教育";
                    break;
                case 3:
                    style="教改动态";
                    break;
                case 4:
                    style="校园新闻";
                    break;
                case 5:
                    style="学院海报";
                    break;
                case 6:
                    style="学生工作";
                    break;
                default:
                    style="";
                    break;
            }
            NewDataBind();
```

```
        }
    }
    private void NewDataBind()
    {
        GridView1.DataSource=new tbNewManager().GetPartNewsByStyle1(style);
        GridView1.DataBind();
    }
    protected void GridView1_RowUpdating(object sender, GridViewUpdateEvent-
    Args e)
    {
        tbNew tbn=new tbNew();
        tbn.ID=Convert.ToInt32((GridView1.Rows[e.RowIndex].FindControl
        ("lblID")as Label).Text);
        tbn.Title=(GridView1.Rows[e.RowIndex].FindControl("txtTitle")as TextBox).Text;
        tbn.IssueDate= Convert.ToDateTime((GridView1.Rows[e.RowIndex].FindControl("
        txtIssueDate")as TextBox).Text);
        if(new tbNewManager().ModifyNew(tbn))
        {
            Page.ClientScript.RegisterStartupScript(this.GetType(), "", "alert('修改成
            功!');", true);
            GridView1.EditIndex=-1;
            NewDataBind();
        }
        else
        {
            Page.ClientScript.RegisterStartupScript(this.GetType(), "", "alert('修改失
            败!'); ", true);
        }
    }
    protected void GridView1_RowDataBound(object sender, GridViewRowEventArgs e)
    {
        if(e.Row.RowType==DataControlRowType.DataRow)
        {
            e.Row.Attributes.Add("onmouseover", "currentcolor = this.style.
            backgroundColor;this.style.backgroundColor='#6699ff'");
            e.Row.Attributes.Add("onmouseout", "this.style.backgroundColor =
            currentcolor");
            //改行的第 4 个单元格,增加删除确认
            e.Row.Cells[4].Attributes.Add("onclick", "return confirm('确认删除吗?')");
        }
    }
    protected void GridView1_RowCancelingEdit(object sender, GridViewCancelEditEve-
    ntArgs e)
    {
```

```
        GridView1.EditIndex=-1;
        NewDataBind();
    }
    protected void GridView1_RowDeleting(object sender, GridViewDeleteEv-
    entArgs e)
    {
        int id=Convert.ToInt32((GridView1.Rows[e.RowIndex].FindControl
        ("LinkButton1")as LinkButton).CommandArgument);
        if(new tbNewManager().DeteteNewById(id))
        {
            Page.ClientScript.RegisterStartupScript(this.GetType(), "", "alert('删除成
            功')", true);
            NewDataBind();
        }
        else
        {
            Page.ClientScript.RegisterStartupScript(this.GetType(), "", "alert('删除失
            败')", true);
        }
    }
    protected void GridView1_RowEditing(object sender, GridViewEditEventArgs e)
    {
        GridView1.EditIndex=e.NewEditIndex;
        NewDataBind();
    }
```

【步骤 5】页面浏览如图 10.7 所示。

3. 新闻添加页面 NewAdd.aspx

【步骤 1】源视图代码如下：

```
<table class="txt" style="width: 483px; height: 303px">
            <tr>
                <td  colspan="3" style="text-align:center">
                    新闻添加『<asp:Label ID="Label1" runat="server" Text="Label"></
                    asp:Label>
                    』</td>
            </tr>
            <tr>
                <td style="width: 66px">
                    新闻标题:</td>
                <td style="width: 324px">
                    <asp:TextBox ID="TextBox1" runat="server" Width="200px" MaxLength
                    ="15"></asp:TextBox> (控制在 15 个字符以内)</td>
                <td style="width: 85px">
                    <asp:RequiredFieldValidator ID="RequiredFieldValidator1" runat="
```



图 10.7 list.aspx 页面

```
        server" ControlToValidate="TextBox1"
            ErrorMessage="* * "></asp:RequiredFieldValidator></td>
</tr>
<tr>
    <td style="width: 66px">
        新闻内容:</td>
    <td style="width: 324px">
        <asp:TextBox ID="TextBox2" runat="server" Height="211px" TextMode
        ="MultiLine" Width="322px"></asp:TextBox></td>
    <td style="width: 85px">
        <asp:RequiredFieldValidator ID="RequiredFieldValidator2" runat="
        server" ControlToValidate="TextBox2"
            ErrorMessage="**"></asp:RequiredFieldValidator></td>
</tr>
<tr>
<td>
图片
</td>
<td>
    <asp:FileUpload ID="FileUpload1" runat="server" />
</td>
</tr>
```

```
            <tr>
                <td  style="width:66px">
                </td>
                <td style="width: 324px;text-align:center">
                    <asp:Button ID="Button1" runat="server" Text="添  加" Width="
                    66px" OnClick="Button1_Click" />
                    <asp:Button ID="Button2" runat="server" CausesValidation= "False"
                    Text="重  置" OnClick="Button2_Click" /></td>
                <td style="width: 85px">
                </td>
            </tr>
        </table>
```

【步骤 2】数据访问层代码如下：

```
public bool InsertNew(tbNew tbn)
        {
            string sql="insert into tbNews(Title,Content,StyleId,IssueDate,imageid)
            values(@Title,@Content,@StyleId,@IssueDate,@imageid)";
            SqlParameter[] para=new SqlParameter[]
          {
              new SqlParameter("@Title",tbn.Title),
              new SqlParameter("@Content",tbn.Content),
              new SqlParameter("@StyleId",tbn.Style.Id),
              new SqlParameter("@IssueDate",tbn.IssueDate),
              new SqlParameter("@imageid",tbn.Imageid)
          };
            int n= DBHelper.ExecuteNonQuery(DBHelper.ConnectionString, CommandType.
            Text, sql, para)>0;
        }
```

【步骤 3】业务逻辑层代码如下：

```
public bool InsertNew(tbNew tbn)
        {
            return new tbNewService().InsertNew(tbn);
        }
```

【步骤 4】表示层代码如下：

```
protected void Page_Load(object sender, EventArgs e)
    {
        if(!IsPostBack)
        {
            int n=Convert.ToInt32(Request.QueryString["id"]);
            switch(n)
            {
```

```
                case 1: Label1.Text="通知公告"; break;
                case 2: Label1.Text="创业教育"; break;
                case 3: Label1.Text="教改动态"; break;
                case 4: Label1.Text="校园新闻"; break;
                case 5: Label1.Text="学院海报"; break;
                case 6: Label1.Text="学生工作"; break;
            }
        }
    }
    protected void Button1_Click(object sender, EventArgs e)
    {
        string strcontent=Server.HtmlEncode(TextBox2.Text);
        strcontent=strcontent.Replace("\r\n", "<br>");
        strcontent=strcontent.Replace("'", "''");
        strcontent=strcontent.Replace(" ", " ");
        string strtitle=Server.HtmlEncode(TextBox1.Text);
        strtitle=strtitle.Replace("\r\n", "<br>");
        strtitle=strtitle.Replace("'", "''");
        strtitle=strtitle.Replace(" ", " ");
        tbNew tbn=new tbNew();
        tbn.Content=strcontent;
        tbn.Title=strtitle;
        tbn.Style=new tbStylManager().GetStyleById(Convert.ToInt32(Request.QueryString
        ["id"]));
        tbn.IssueDate=DateTime.Now.ToLocalTime();
        string FileName=this.FileUpload1.FileName;
        if(FileName.Trim().Trim().Length !=0)
        {
            int n=FileName.LastIndexOf(".");
            tbn.Imageid=FileName.Substring(0, n);
            string strpath=Server.MapPath("../image/NewConvers/"+FileName);
            FileUpload1.PostedFile.SaveAs(strpath);          //把图片保存在此路径中
        }
        if(new tbNewManager().InsertNew(tbn))
        {
            Page.ClientScript.RegisterStartupScript(this.GetType(), "", "alert('插入成
            功!'); ", true);
        }
        else
        {
            Page.ClientScript.RegisterStartupScript(this.GetType(), "", "alert('插入失
            败!'); ", true);
        }
    }
```

```
protected void Button2_Click(object sender, EventArgs e)
{
    TextBox1.Text="";
    TextBox2.Text="";
}
```

【步骤 5】页面浏览如图 10.8 所示。



图 10.8　NewAdd 页面

参 考 文 献

[1] 郑广成,沈蕴梅等. C#程序设计项目化教程.北京：中国水利水电出版社,2012.

[2] 北京阿博泰克北大青鸟信息技术有限公司.使用 ASP.NET 技术开发网上书店.北京：科学技术文献出版社,2011.

[3] 北京阿博泰克北大青鸟信息技术有限公司.开发基于 Ajax 和控件技术的 Web 应用系统.北京：科学技术文献出版社,2011.

[4] 王学卿,孙伟,郑广成. 动态 Web 开发技术—ASP.NET.北京：中国铁道出版社,2009.

[5] 王淑敏. ASP.NET 动态网站设计. 北京：清华大学出版社,2010.